7N-63-TM
48034

ARTIFICIAL INTELLIGENCE RESEARCH BRANCH

1991 Progress Report and Future Plans

Report FIA-91-3-03-1
April 1991

NASA
Mail Stop 244-17
Ames Research Center
Moffett Field, CA 94035

The Artificial Intelligence Branch 1991 Progress Report was a product of the entire Branch. Peter Friedland, Branch Chief, and Monte Zweben, Assistant Branch Chief, had primary responsibility for the document's organization and content. Helen Stewart served as technical editor. Pat Langley led the writing of those sections describing our external research collaborators; Hamid Berenji, Mark Drummond, and Monte Zweben assisted in this effort. All project leaders contributed substantially to the individual descriptions of inhouse research and projects.

Table of Contents

1. Introduction

This document describes the progress and plans of the Artificial Intelligence Research Branch (FIA) at the NASA Ames Research Center (ARC) in 1991. Activities span a range from basic scientific research through engineering development to fielded NASA applications, particularly those applications that are enabled by basic research carried out in FIA. Work is conducted in-house and through collaborative partners in academia and industry. Our major focus is on a limited number of research themes with a dual commitment to technical excellence and proven applicability to NASA short, medium, and long-term problems. FIA acts as the Agency's lead organization for research aspects of artificial intelligence, working closely with a second research laboratory at the Jet Propulsion Laboratory (JPL) and AI applications groups at all NASA centers.

There are currently forty-nine members of the technical staff in FIA, with 18 holding a PhD in computer science or related fields and an additional 21 holding other advanced degrees. FIA staff participate actively in both the artificial intelligence and aerospace professional communities, serving as editors and editorial board members of several major journals, as members of the program committee of the National Conference on Artificial Intelligence, and as members of Technical Committees of the American Institute of Aeronautics and Astronautics (AIAA). Dr. Peter Friedland is the Chief of FIA and Mr. Monte Zweben is the Assistant Chief.

Our program is organized by thrusts which are designed to address different classes of NASA problems:

- Mission Operation Assistance: the development of AI-based systems that can act as "intelligence amplifiers" for humans on the ground and in space who have the responsibility to conduct mission operations.

- Data Analysis Techniques: the development of tools to improve the analysis of vast amounts of science and engineering data resulting from NASA missions.

- Autonomous Control: the development of in-the-loop systems for planning actions, monitoring, controlling, and diagnosing current and future spaceborne systems.

- Knowledge Base Technology: the development of mechanisms for acquiring, combining, maintaining, and utilizing knowledge relating to the devices NASA designs, builds, and operates over long life-spans.

Technically, several major research themes cross all of the functional domains. These are:

- Planning and Scheduling: deciding on a sequence of actions to achieve a set of complex goals and determining how to allocate resources to carry out those actions.

- Machine Learning: techniques for forming theories about natural and man-made phenomena; and for improving the problem-solving performance of computational systems over time.

- Construction, Maintenance, and Utilization of Large-Scale Knowledge Bases: research on knowledge acquisition, knowledge representation, combination of knowledge from many different sources, and multi-purpose utilization of knowledge bases large enough to represent substantial parts of complex NASA systems such as the Space Shuttle and Space Station Freedom.

- Human-Machine Aspects of AI-Based Systems: many of the current and future applications of artificial intelligence in NASA involve a strong interaction with highly trained humans (scientists, engineers, astronauts, etc.). Ensuring the ability to communicate with those humans in an effective manner is vital to eventual user acceptance of all of the other research themes.

Our work is funded by a variety of sources from NASA and other federal agencies. The single largest sponsor of work is the Artificial Intelligence Program of the Information Science and Human Factors Division of the NASA Office of Aeronautics, Exploration, and Technology (OAET). (Dr. Mel Montemerlo is the manager of this program). We also receive support from the Space Station Freedom Advanced Development Program for the PI-in-a-box Project and from the Office of Space Science and Applications Information Systems Division for the Scientific Model Building Project. (Mr. Mark Gersh is the responsible program manager.) The Defense Advanced Research Projects Agency's Information Science and Technology Office (DARPA/ISTO) (Major Steve Cross is program manager) co-funds our work in Intelligent Interacting Agents.

2 Mission Operations Assistance

2.1 Automated Scheduling Tools

2.1.1 Space Shuttle Ground Processing Scheduling

Monte Zweben , Eugene Davis, Ellen Drascher,Todd Stock
Robert Gargan, Michael Deale, Brian Daun; Lockheed AI Center
Cindy Mollakarimi, Danielle Schnitzius, Mark Yvanovich; Lockheed Space Operations Company

The Space Shuttle Ground Processing Scheduling project is deploying a constraint-based scheduling system at the Kennedy Space Center (KSC). The goal of the project is to assist KSC management in streamlining Shuttle processing. The KSC flow management team has the responsibility for preparing the Space Shuttle for flight. Shuttle processing (referred to as the *flow*) actually begins while the orbiter is in space. The Johnson Space Center Mission Control team reports subsystem performance information and anomalies to KSC that must be investigated when the orbiter returns to Earth. The flow management team also meets the orbiter at its landing site and performs a structural inspection of the orbiter with the hope of making an accurate assessment of any damage that might have occurred during the mission. Based on this early assessment, the flow management team modifies its ground processing schedule accordingly. The first activity of the flow is the orbiter's arrival at the KSC Orbiter Processing Facility (OPF). There, each orbiter subsystem is inspected, repaired, and tested. The OPF is a large hangar capable of housing two orbiters. Processing in the OPF is labor-intensive and hazardous. Many volatile materials and large objects are manipulated. This process usually extends for about 65 days, after which the orbiter is transported to the Vehicle Assembly Building (VAB). There, the orbiter is mated with the Solid Rocket Boosters and the External Tank. Subsequently, the vehicle (i.e., orbiter, boosters, and tank) is transported to the Launch Pad for final launch preparations.

The processing within the OPF is comprised of a set of extremely dynamic and complex activities coordinated by the flow management. Execution of a ground processing schedule is plagued by the uncertainty inherent in most complex operations environments. For example, delays are caused by backordered parts, contention for resources, and competition for conflicting configurations of the orbiter. Additionally, equipment often malfunctions and personnel get sick. The uncertainty of this environment is heightened by the fact that it is impossible to completely plan the processing activities in advance because unpredictable problems are detected during inspection tasks. As a result, dynamic rescheduling is a crucial aspect of Space Shuttle processing.

In 1991, the Ames Artificial Intelligence Research Branch, the Lockheed Artificial Intelligence Center, and the Lockheed Space Operations Company have jointly attacked this dynamic rescheduling problem. Gerry, the Ames constraint-based scheduling system, has been extended to support the KSC ground processing problem. There have been three major modifications to Gerry that support the KSC application. First and most important, a new search algorithm, called constraint-based simulated annealing, has been developed. Second, the algorithm has been extended to deal with the fixed-preemptive scheduling problem of which ground processing is an instance. And, finally, the system has been extended to represent state information. The attributes of the "environment" that change over time are represented as state information. This includes the configuration of the orbiter as well as the spatial zones within the OPF. Orbiter configuration includes the positions of the landing gear, payload bay doors, elevons, rudder speed brake, and status of the orbiter power and hydraulics systems.

Constraint-based simulated annealing is a search technique that iteratively improves complete schedules. Scheduling problems are specified as a set of tasks, resources, and state attributes that are related to each other by constraints. Generally, there are three classes of constraints: temporal, capacity, and state. Temporal constraints relate the start and end times of tasks. The most prevalent temporal constraint is the prerequisite type that requires the start of a task to be equal to or follow the end time of its prerequisites. Capacity constraints declare that a resource used by a task must not be over allocated. State constraints require attributes to have a certain values over time. Examples of state constraints include: doors in certain positions, power being on or off, and the spatial congestion of an area being below a threshold. For each constraint, there is an associated penalty function and a repair function. The penalty function returns how badly the constraint is violated and the repair function returns a modified schedule. This new schedule is intended to lower the constraint's penalty. The basic scheduling/rescheduling algorithm repairs highly penalized constraints until the user interrupts the process or until an acceptable schedule is found. The schedule is scored by a "cost" function which is a function of all the constraint penalties. The main idea behind annealing is that it usually rejects schedule modifications that are worse (i.e., have higher cost functions) than the current one, but occasionally randomly accepts them in order to avoid getting stuck in local minima. In other words, if no repair immediately improves the schedule, the system will not stop -- it will accept a repair that temporarily worsens the schedule with the hope that subsequent iterations will show a net improvement. The main contribution of this work is the knowledge-based framework that enables quick convergence to solutions. Additionally, the system has an "anytime" nature in that it can be interrupted at any point and the best solution at that time is returned.

In late 1990 and early 1991, Gerry has been adapted to address fixed-preemptive scheduling problems. Fixed-preemptive scheduling problems differ from classical problems in that activities must be split up into subtasks, each with its own resource and and state information, according to a pre-specified task calendar. The task calendar typically models labor schedules such as first shift on weekdays and no weekends, or everyday including holidays. Instead of an absolute duration, tasks are associated with work durations that specify the aggregate amount of activity required to complete the task. This work duration is always equal to the sum of the absolute durations of the subtasks for a given task. Any time a task is moved on a schedule, its split subtasks must be retracted and the task must be re-split. This is computationally costly because each split has resource and state effects that must be retracted. Resource availability and state information are implemented as linked lists, and therefore, each retraction (and reassertion) is an operation with worst case complexity O(N), where N is the average number of transitions on a resource pool or state variable. Because of this cost, we developed an indexing scheme that significantly reduces the average case complexity of the O(N) operations. We discretize time into buckets and each bucket points to the general area of that time in the linked list. For any list operation, a simple lookup operation returns the bucket and then the list is searched from that point on (instead of from the beginning). Another problem imposed by fixed-preemptive scheduling is storage management. Originally, the implementation allocated new split tasks after each task move. The overhead caused by allocating new CommonLISP objects (i.e., CLOS objects) and then garbage collecting them, overwhelmed the system. Consequently, we implemented our own storage management scheme that avoids allocating new objects and never throws them away.

In summary, scaling our research prototype to a useable end-product has required extensive, unpredicted modifications. First, O(N) data structures had to be improved because N ranged from 1000 to 6000. Second, our CommonLISP storage management and garbage collection implementations were too slow. We believe these obstacles have now been overcome.

As mentioned previously, state information can now be modeled within Gerry. While resource contention is a frequent cause of rescheduling at KSC, conflicting orbiter configuration requirements also cause rescheduling. For example, suppose that a task which requires the payload bay doors to be closed has been delayed until Tuesday because certain tiles have been backordered. However, the KU-Band Antenna deployment test is also scheduled on Tuesday. This is a problem because the KU-Band antenna is extended out from the payload bay, and therefore, requires the doors to be open to the 160 degrees position. If the doors were closed, the antenna would be damaged. The interesting aspect of this interaction is that these two activities have no causal relationship, and therefore, there are no prerequisite constraints defined between them. If we could not model the effects and requirements on state attributes, it is feasible that a flawed schedule could be produced that would require substantial human modification. We have extended Gerry with a state attribute language that allows one to specify how multiple effects combine and persist. For example, if two tasks raise a temperature at the same time, then their effects are aggregated. However, other effects do not combine but rather supersede each other. If a task turns a switch on and another turns it off, the later effect will "clip" or supersede the earlier one.

In 1991, an extensive knowledge engineering effort is underway to extract information from the flow management team, the orbiter engineers, and the orbiter technician organizations. This has been a long and somewhat painful process because most of the required data is not in electronic form. Additionally, it has been difficult to get accurate estimates of resource capacity and usage quantities from all the different organizations involved. Task data has also been corrupted. Artificial temporal constraints have been added to compensate for the lack of resource and state modeling in existing scheduling tools. For example, if it is known that two tasks interfere "usually" because of a resource contention, one will be required to follow the other. If more resources are acquired later on, or if rescheduling jumbles the schedule, this constraint is likely to remain and artificially restrict the schedule! Determining which constraints are causally valid and which are experientially added will be an ongoing activity.

Our plan for 1992 is to shadow two flows of the orbiter Atlantis and at least one flow of the orbiter Columbia. Most important, we hope to fully support the maiden flow of Endeavour by tracking the flow with respect to resource utilization. By the end of 1992 we expect to have all major orbiter configurations modeled. We have also begun the implementation of a novel data entry system for orbiter configuration. Using optical character recognition software and a scanner, our system will update its expectation of state conditions after processing a checksheet that the flow management team currently completes during OPF "walkdowns".

We also expect to optimize and streamline the implementation. One of the major efforts will be the implementation of a locality technique (motivated by Lansky's GEMPLAN; see Section 4.3.1) to reduce the amount of constraint checking during rescheduling. Regions of locality will help split the schedule into pieces and only those pieces affected during rescheduling will be considered in the constraint test and repair process. Regions will correspond to temporal "chunks" of time, spatial zones within the OPF, and possibly activities associated with bottleneck resources. See Section 2.1.2 for a discussion on acquiring these regions automatically. We will also improve the functionality of the "snapshot" mechanism that will facilitate what-if analysis and enable interruptions.

Another major effort will be the acquisition and deployment of domain-specific rescheduling knowledge. Undoubtedly, our domain-independent, heuristic repairs will conflict with the experience accumulated by flow management. Therefore, this experience must be representable in our system and usually must supersede the domain-independent knowledge. In 1992 we will acquire this knowledge, create a rule-based framework for representing this knowledge, and finally integrate this knowledge into the annealing framework.

We also plan to explore optimization techniques. Schedules in the OPF are modified reactively with little time for optimization. Our system will be augmented with global optimization constraints and repair strategies that will allow the flow to be improved. Examples of optimization constraints include overall flow time, due date tardiness, third shift activity, and the number of times expensive tasks like power ups and full hydraulic tasks are repeated. The repairs for these constraints will include moving activities earlier and removing tasks from the schedule.

Milestones

1991:	- "Shadow" scheduling of Atlantis orbiter processing
	- Completion of STS-37 knowledge engineering effort
	- Paper to be presented at the Twenty-Eigth Space Congress
	- Schedules from system uploaded to SPDMS-II
1992:	- Operational use of system by flow management assisted by LSOC personnel
	- Optimization criteria added to system will improve overall flow.
1993:	- Distributed access to scheduling data by KSC community
	- System used by flow management autonomously.
1994:	- System used routinely during scheduling meetings.

2.1.2 Machine Learning to Improve Planning and Scheduling

Steven Minton, Monte Zweben , Andy Phillips, Gene Davis

The primary goal of this work is to improve upon current AI-based methods for planning and scheduling. We are focusing on two related approaches. First, we are analyzing and comparing current planning and scheduling techniques, with the aim of developing better heuristic methods than currently exist. Second, we are designing machine learning techniques that will automatically improve the performance of planning and scheduling systems over time.

We are continuing our work with the min-conflicts scheduling heuristic which appears promising for a variety of scheduling applications. This heuristic guides the scheduling process by minimizing the total number of constraint violations in a schedule. We are analyzing why the technique works well in certain application environments in order to validate and refine the theory upon which it is based. This work has already been used by researchers at the Space Telescope institute for improving the Hubble Space Telescope Scheduler and we are continuing our collaboration on the telescope scheduling project. We are also working on extending this application to other automated telescope projects. In addition, we are analyzing other planning and scheduling heuristics in order to develop theories which will allow them to be more generally applied as well. One heuristic, often referred to as non-linear planning, has recently been the focus of our work, and we are in the process of publishing a new analysis of this heuristic.

The second primary thrust of our work is to apply machine learning techniques to planning and scheduling applications, including the telescope application mentioned above, as well as other applications such as space shuttle payload processing. We are primarily investigating analytic learning techniques, such as Explanation-Based Learning (EBL), which generalize specific "training instances" by proving properties about the instances. The learning method operates as follows. First the system proves that the instance has a certain property of interest. (The proof is referred to as an "explanation"). Then, after the proof is generated, the system finds the weakest conditions under which the proof holds. These weakest conditions constitute a generalization of the training instance. All instances which are described by the generalization are guaranteed to have the aforementioned property.

We are currently extending previous work on Explanation-Based Learning (EBL) to new planning and scheduling tasks, and developing theories which describe the circumstances under which EBL provides useful performance optimizations. One recent development is the integration of EBL with methods for abstraction (a commonly used technique in AI planning systems). This allows both abstraction and EBL to be used in conjunction with each other.

In 1991, we developed Plausible Explanation-Based Learning (PEBL) which enabled EBL to exploit multiple examples. PEBL simply caches explanations and uses multiple examples to confirm that the explanations repeatedly apply. PEBL was tested on the Gerry constraint-based scheduling system to learn resource bottlenecks. Backtracking in Gerry invoked the PEBL process which conjectured that certain resources were bottlenecks that caused the backtracking. Then some these explanations were confirmed with further examples until a threshold of confirmations were acquired. At this point, search control rules were synthesized that advised Gerry to assign these bottleneck resources before assigning start and end times. This strategy led to 30%-40% performance savings but at the cost of sometimes degrading the quality of the schedules. More specifically, when compared to Gerry without learned knowledge, the schedules that used learned knowledge sometimes had longer work-in-process times.

We will generalize PEBL to represent the explicit uncertainty associated with any plausible explanation. A PEBL domain theory domain theory contains both normal and defeasible horn clauses. The defeasible horn clauses are treated as material implications during the explanation process, but have a conditional probability associated with them. Statistics are gathered over multiple examples to update these conditional probabilities. Then, the uncertainty of an explanation is calculated using a Bayes Net that combines these conditional probabilities to form a single measure of belief. When the system has substantial belief in an explanation and a sufficient number of samples are observed, it then determines whether the learned knowledge is likely to be useful given its match cost and expected impact on solution quality. If so, the system incorporates the learned knowledge into the performance engine's knowledge base. PEBL extends previous attempts to equip EBL with an empirical component because it uses an explicit measure of uncertainty. This measure allows PEBL to handle "noisy" domains. Other methods can get "tricked" by noise.

We also plan to investigate a locality technique (motivated by Lansky's GEMPLAN) to reduce the amount of constraint checking in the Gerry rescheduling system. Regions will correspond to temporal "chunks" of time, spatial zones within physical facilities, and possibly activities associated with bottleneck resources. The most contentious regions of the schedule should be focused upon because they will be the hardest to solve. Region definitions can be acquired with experience. For example, the system might notice that a particular week is extremely difficult to schedule or that a particular resource is troublesome (e.g., the PEBL experiments described above). We will explore different region definitions, measure their utility, and explore automatic region learning methods.

Milestones

1991: - AAAI-91 Paper on integrating abstraction with EBL.
- Demonstrate machine learning component of the Gerry scheduling system on a complete STS orbiter processing data set.

1992: - Complete a scheduler for a ground-based automatic telescope.
- Machine Learning Workshop paper submitted on the generalization of PEBL with a Bayesian component.
- First results on learning localities within the KSC ground processing domain.
- First results on learning to improve iterative improvement searches:

1993: - Learning algorithms integrated into deployed KSC application.

2.1.3 Improving Search-Based AI Systems

Thomas G. Dietterich; Oregon State University

The central issues addressed in this research project are the discovery of new abstract objects and operators and their use in improving the efficiency of problem solving. The focus is on solving optimization problems such as occur in minimax search, scheduling, and engineering design. Problem solving in such domains can be viewed as search for some state that satisfies (to a greater or lesser extent) a set of constraints, and these constraints can be used to direct the learning process.

Initial work by Flann and Dietterich in 1990 and 1991 focused on chess endgames. They developed an abstract domain theory that let them apply explanation-based learning techniques to extract general sufficient conditions for achieving a given goal (say, avoiding checkmate) from search trees for two-person games. Their approach uses these sufficient conditions to define new abstract objects and operators that ignore irrelevant details and thus lead to more rapid problem solving. They have also demonstrated a method for compiling these concepts into extremely fast recognition rules by exploiting geometrical knowledge of the domain. The result was a speedup in problem-solving efficiency of more than five orders of magnitude. This improvement is significantly greater than that observed in previous work on learning in problem solving.

In a related project, Cerbone and Dietterich have addressed the task of two-dimensional structural optimization in engineering design. In 1991, they implemented a technique for discovering optimal design rules from successful designs. The approach involves first solving a set of simple problems using traditional numerical optimization techniques. To increase the speed of this inherently slow process, Dietterich has used symbolic transformation and simplification techniques to obtain more efficient closed-form expressions. He predicts such methods can lead to 50% speedup, but that more substantial improvements can result by reducing the dimensionality of the search space.

Dietterich's approach to this problem, started in 1990 and being pursued in 1991, is to analyze problems solutions in order to formulate condition-action rules that let the problem-solving system construct the optimal solutions directly from the given problem parameters. This analytic process employs abductive reasoning, a method that constructs explanations that include default assumptions. For example, two angles in a structural design may be equal, and the method could assume this was necessary even though it could not prove this connection. The resulting rule is incorporated into the optimization search, reducing the number of variables that must be considered. Hand simulations of this technique produced optimization rules that were previously unknown to the researchers. Work in 1992 will focus on using inductive methods to determine the range of applicability for such rules.

More recent work, just beginning in 1991 and continuing in 1992, focuses on learning in another domain involving optimization -- scheduling. Dietterich and colleagues have implemented a resource-based scheduler and carried out initial tests on actual problems that have occurred in scheduling the space shuttle cargo. Within this framework, they have started to explore a number of methods for improving the scheduling process through learning, including a technique for automatically deriving admissible search heuristics. Future work will develop and evaluate these alternative approaches, with the long-term goal of aiding the scheduling of actual NASA missions.

Milestones

1991: - Refine methods for chess learning and evaluate them through systematic experimentation.
- Implement method for learning in 2-D structural design and carry out initial evaluation.
- Begin adapting the use of search improvement techniques on a NASA scheduling problem; implement initial scheduler.

1992: - Complete research on speedup learning in chess, attempting to generalize the results for application to other domains.
- Extend method for learning in 2-D structural design to induce applicability conditions for optimization rules.
- Implement initial system that improves its ability to generate scheduling and begin systematic experiments in the domain of space shuttle cargo schedules.

1993: - Adapt the design learning system to handle complex problems in 2-D structural design, using such tasks to evaluate the system's ability to improve with experience.
- Improve system for learning in scheduling domains, based on results of initial experiments in NASA domains.

1994: - Complete research on learning in 2-D structural design and begin search for NASA domains in which this method can be applied.
- Continue improvement of the learning scheduler and begin adapting the system for field tests.

2.1.4 Constraint-Directed Scheduling and Planning of Space Missions

Stephen Smith, Nichola Muscettola, Carnegie Mellon University

The objective of this project is to develop representations and problem solvers for the planning and scheduling of space missions in the presence of complex physical constraints. This work builds upon previous work on the ISIS and OPIS factory scheduling systems within which knowledge about scheduling constraints direct the search process. These systems allocate resources to competing activities over time, in the presence of conflicting objectives and preferences. These techniques have now been extended to cleanly integrate resource allocation with plan generation capabilities for a complex physical system.

The Hubble Space Telescope (HST) short-term observation problem is the NASA domain that drives this research. The project team works closely with the Space Telescope Science Institute (STScI) to acquire accurate constraint knowledge and preferences for their system and to gather feedback on their developments.

An enhanced planning and scheduling system named HSTS was completed and demonstrated at Ames. Currently, two astronomical instruments, the data buffers, the data transmitters, and the local tape recorder of the HST are modeled within the HSTS system . The philosophy of the project is to build complete planning and scheduling systems for increasingly realistic models of the HST. The current system is comprised of a domain description language, a temporal database, and a planning/scheduling search engine. The domain description language enables one to specify the legal configurations and transitions of state variables of the system. Further, it allows one to enforce metric time bounds on these constraints. For example, an instrument must be in a ready mode before executing an observation and the observation must not exceed a certain amount of time since the last calibration. All the complex physical and temporal relationships between the elements of the spacecraft are specified in the domain description language.

The temporal database, which is similar to Dean's Time Map Manager, operationalizes the domain description language by maintaining only those behaviors that are consistent with respect to the domain specification and the commitments made by the search process. By capitalizing on both domain-dependent and domain-independent heuristics, the search framework makes several types of choices. It refines abstract problems into more specific problems, it decomposes problems into subproblems (at the same level of abstraction), it selects among the mutually exclusive choices specified in the domain description language, and finally, it propagates temporal constraints. The heuristics of the current search process generally attempt to maximize scientific return by reducing idle time. Preliminary experiments with HSTS reveal promising results in terms of both scheduling performance and schedule optimality. The architecture is one of the few successful attempts to integrate planning and scheduling.

The search heuristics in the current HSTS are local and greedy, which is likely to be insufficient once the HST model is made more complex. New constraint-directed strategies will be incorporated into HSTS to help structure the process more globally, thus improving overall optimality. These strategies will exploit the structure of the evolving solution space similar to the techniques used in other CMU research on scheduling. For example, resource bottlenecks will be sought that suggest certain variables should be addressed before others. The HST model will be made more complex with more instruments and new constraints. CMU and STScI will together determine whether the existing HSTS framework or concepts can be utilized operationally by the HST program, and CMU and Ames will together investigate whether other astronomical programs can make use of the software or ideas developed within this effort.

Another focus of 1992 will be abstraction. It is unclear how much effort should be expended at various levels of abstraction while scheduling. For example, too little effort at higher levels could lead to expensive backtracking at the lower levels but too much thought at the higher levels could also be overkill. Extensive experimentation of HSTS will help analyze this tradeoff.

Milestones

1991:	- HST model enhanced
	- HSTS prototype demonstrated
1992:	- Abstraction experimentation complete
1994:	- HSTS operationally deployed

2.2 Advanced Interaction Media

2.2.1 Computer Integrated Documentation

Guy Boy, Jody Gevins, Bharathi Raghavan
Irv Statler, Ames Aerospace Human Factors Division
Cécile Paris, University of Southern California-ISI

Computer Integrated Documentation (CID) integrates hypermedia and knowledge-based systems. A first prototype of a CID system has been developed and demonstrated this year, showing the value of context-dependent indexing and information retrieval. Hypertext is a good tool for the development of documentation systems. Hypertext systems increase accessibility, but they do not provide any built-in selectivity mechanism. In other words, while non-linear or hypertext systems may dramatically increase the accessibility of information, this increased accessibility may magnify an already severe problem of selection. For these reasons, our knowledge-based systems technology can be very helpful in alleviating the selection problem and the cognitive overhead of the user.

The CID project has concentrated on four major activities: (1) the development of a generic hypermedia system augmented by an intelligent indexing and information retrieval mechanism; (2) the study of incremental context acquisition in information retrieval; (3) the application of machine learning to context-sensitive indexing; and (4) the application and first evaluation of the CID tool on the Program Requirement Document for the Space Station Freedom (SSF).

The CID system being developed contains: a knowledge representation methodology, called "Block KR;" a knowledge acquisition system that is integrated with the Block KR; and, a user interface generator. In 1990, the Block KR was formalized as a representation framework for operation manuals, procedures checklists, and other on-line tools useful for controlling and repairing complex dynamic systems. In 1991, we have efficiently implemented it in the CID system to represent links between descriptors and referents. To automatically help the user, the problem is to "contextify" the links between documentation nodes, that is to provide relations between descriptors and referents that are valid in the current context. These relations vary depending on the situation and the user. This context reduces the number of possible referents for a descriptor that a user has to consider, and thus speeds his or her search. An algorithm called IARC (Index Acquisition and Refinement according to Context) has been developed. We define the context acquisition problem as discovery of "abnormal conditions" and generation of recovery actions, as well as reinforcement of current actions. Our system observes the user's actions during the browsing tasks, and, by noting whether a specific referent was considered a success or failure by the user in a particular context, it is able to refine the indices to reflect their context of use. In this way, our system automatically acquires the knowledge necessary to operationalize a user model: which indices are appropriate when, and for which user.

The current system has been tested on a small but significant subset of SSF documentation (a Technical Memorandum is available describing these results in detail). Besides providing an intelligent interface for browsing large documents, the ability of CID to acquire automatically the context in which strategies are appropriate is significant. First, it allows the system to provide a *tailorable* browsing facility. Indeed, CID will learn which referents are to be presented for which user. Second, it shows that it is feasible to immediately incorporate the user's feedback into the CID's knowledge, with the possibility of thus improving the system's performance. In this way, there is no need to collect and analyze large amount of information about how users interact with the system to later incorporate (manually) in the program. The system performs this task itself.

Work will continue on improving the block KR and associated knowledge acquisition techniques to provide effective performance on increasingly larger human-machine interaction problems. The CID system itself will be evaluated in the context of the Space Station Freedom Program Requirement Document, Communication Procedures in the Mission Control Room, and the RECON electronic library system. Collaboration will begin with Cecille Paris of USC-ISI and Tom Rindfleisch of Stanford University where similar problems are being addressed in different domains.

Several issues will be addressed. First, formalization of the contextual conditions is still problematic. On the one hand, the contextual conditions should be minimal to avoid excessive calculations. On the other hand, they must include as much information as possible to characterize the current situation. Second, we are still trying to define the "ideal" threshold for deciding to create a new block, delete an "unused" one, or include an action in a block. This will be done by empirical testing on all of the documentation sets described above.

Milestones

1991: - Demonstrate CID on JSC MCC documentation.
- Develop a context clustering algorithm
- Utilize machine learning methods to implement adaptive and context-sensitive indexing.
- Present formalization of context acquisition at IJCAI and AAAI.

1992: - Demonstrate integration of CID with SSF TMIS.

1994: - CID fully integrated and operational for NASA documentation management and maintenance.

2.2.2 Procedure Management and Maintenance

Guy Boy
Steve Ellis, Rick Jacoby; Ames Aerospace Human Factors Division
Robert Mah, Jay Steele; Ames Intelligent Systems Technology Branch
Nathalie Mathé; visiting ESA Fellow

The goal of this project is to improve the performance of teleoperation in complex dynamic environments. Our approach takes an alternative view to conventional planning: the design of procedures (plans) is based not only on the way teleoperated systems has been built, but also on user information requirements and suggestions when they are operating the systems. When starting to operate a particular system, a person learns formal procedures that facilitate subsequent interaction with the teleoperated system. These procedures are then improved over time, leading to incrementally more operational knowledge. Such a knowledge compilation mechanism is an important research issue.

Procedures are represented using the same block KR as in the CID project. A knowledge block includes five components: contextual conditions, triggering preconditions, a set of actions and attached abnormal conditions, and a goal. This representation has already been developed and used successfully in a telerobotic application by Nathalie Mathé when she was working with CNES in France. The block representation also supports a learning mechanism that is based on acquisition of new blocks and refinement of existing blocks by adding or reinforcing abnormal conditions and contextual conditions. As in the CID project, contextual and abnormal conditions will be acquired through experimentation. This research effort is an extension of the work already done in the documentation project from a static environment to a dynamic environment.

A major problem in using "canned" procedures is the mode of communication between a human operator and a teleoperated intelligent system. A technique for solving this problem is to first apply a procedure in a simulated or "virtual" environment and only provide actual commands to an intelligent system when the operator is satisfied with the virtual results. Since the real world may not always match the virtual world, a feedback mechanism is needed to warn the operator of abnormal or unexpected results in command execution. Recovery strategies can be tested in the virtual environment and then applied to the actual intelligent system. In 1991, a Puma robot arm performing simple construction tasks has been used as an experimental testbed, a model of this robot has been implemented in the virtual environment, and a first mockup of procedure management and maintenance (PMM) has been tested separately. Next year's goal is to integrate the three components to solve proximity operations as well as telerobotics problems. We will develop a user-friendly procedure acquisition interface that allows the system to create and modify blocks from user's suggestions. This interactive knowledge acquisition by experimentation will provide successful fault recovery strategies for later use in analogous situations. A generalization mechanism will be studied to improve the structure of the block base, and the search at performance time.

Milestones

1991: - Complete knowledge acquisition and machine learning methods to generalize procedures and design an analogical mechanism for handling procedures
1992: - Demonstrate integration of virtual reality devices with PMM on an integrated testbed

3 Data Analysis Techniques

3.1 PI-In-a-Box

Silvano Colombano, Michael Compton, Richard Frainier
Irv Statler, Ames Aerospace Human Factors Division
Chih-Chaw Lam, Stanford
Larry Young, MIT

The goal of the Principal Investigator in a Box project is to help improve the quality of the scientific results of experiments conducted in space. Our approach is to develop an "intelligent assistant" for astronauts performing space-borne science. The system is designed to assist in the collection and analysis of experiment data, recognition of "interesting" data, management of the experiment protocol, and diagnosis and troubleshooting of the experiment apparatus. The project team's efforts this year focused primarily on bringing together the various components into a functioning system and using it to help collect actual data from the "Rotating Dome", an experiment in vestibular physiology devised by Professor Laurence Young of MIT, in preparation for the first Spacelab Life Sciences Mission (SLS-1) currently scheduled for May, 1991.

A "Data Computer", that supports the data collection and quality monitoring modules, has been successfully integrated with an "AI Computer" that supports the data analysis, protocol management, and diagnostic modules. Together, these computers enable the modules to efficiently process real data while it is collected from astronaut subjects. An Executive Module, also completed this year, mediates communication between the various modules. A Diagnosis and Troubleshooting Module was also implemented, and work is underway to integrate it with the system.

A new user interface was developed and integrated with the system this year. This interface, developed by the Human-Computer Interaction Laboratory at Johnson Space Center and the Aerospace Human Factors Research Division at Ames, was developed to improve the usability and understandability of the system by astronaut users.

The system was used on the ground to help collect "baseline data" for the dome experiment in preparation for SLS-1. Ground baseline data is typically collected at various times before a mission (launch-minus-150 days, launch-minus-75 days, launch-minus-45 days, and launch-minus-15 days), and immediately after the mission. The PI-in-a-Box system was used to collect pre-flight data at the first SLS-1 baseline data collection sessions in August, 1990 and January, 1991. The system will continue to be used during subsequent SLS-1 baseline data collection sessions, and on the ground during the SLS-1 mission in May. The plan is to then have the system used by astronauts in flight during the SLS-2 mission in 1993.

The ground version of PI-in-a-Box is already proving to be useful. During the January baseline data collection session, the Data Quality Monitor detected problems in the incoming data when the operator was too busy to notice, which is a promising sign for what might happen on SLS-2. In addition, the user interface facilitated the creation of ground protocols. An operation that used to require a couple of hours was reduced to a few minutes.

Two potentially reusable software modules were developed that may be of interest to other software developers. The first is a mechanism for "seamless" data sharing between CLIPS (a NASA-developed expert system shell) and HyperCard (a user-interface development and programming environment for the Apple Macintosh). This mechanism allows applications written in these two tools to efficiently pass data back and forth. The second software module is a CLIPS-based relational data base management system, which allows applications written in CLIPS to efficiently store and retrieve homogeneous data elements. Although still in its early stages, this module will probably improve the organization and robustness of the PI-in-a-Box system and is likely to be applicable to a broad class of CLIPS-based applications.

A related effort begun in 1991 is the development of a PI-in-a-Box "demo disk", which will be a self-contained simulation of the system designed to help advocate the project and solicit feedback about the functionality of the system. The initial release of the demo disk is currently scheduled for mid-March, 1991.

Project efforts during 1992 will focus primarily on preparation for flight of the system during the SLS-2 mission, currently scheduled for May, 1993. This will involve final refinements of the system (based on experience during SLS-1), and re-hosting of the system on flight-qualified hardware. The system will also be used during baseline data collection sessions for SLS-2.

In 1992, the PI-in-a-Box team will also begin generalization of the system. This will involve investigation and selection of another experiment, and refinement of the various system modules to support the data collection and analysis requirements of that experiment. These efforts will result in a more general "Astronaut Science Advisor."

Milestones:
1991: - Ground test during SLS-1 flight (mission postponed from previous 8/90 date)
- Initial version of PI-in-a-Box "demo disk"
- Final integration of the Diagnostic and Troubleshooting Module
1992: - Completion of flight hardware and software for SLS-2
- Generalization of system and application to another experiment
1993: - Flight of PI-in-a-Box system aboard SLS-2 (mission postponed from previous 8/92 date)

3.2 Automatic Classification and Data Analysis

3.2.1 AutoClass

Peter Cheeseman, John Stutz, Robin Hanson

In previous years, we developed the basic theory of how to use Bayesian methods for automatic data analysis by classification. We implemented this theory as the AutoClass series of programs, with AutoClass II completed in 1989 and AutoClass III in early 1991. Many collaborators from government, academia, industry have used the latest version; some of the more interesting results are discussed below. AutoClass can find classes where the data consists of cases described by discrete and/real-valued data. The software generates a report describing the classes it found, and these classes are guaranteed not to be an artifact of the search process.

The largest test, to date, of AutoClass was to the IRAS Low Resolution Spectral database containing the infrared spectra of over 5,500 stars. The resulting classification included many previously known spectral classes, as well as new classes of subtly different spectra, since confirmed by independent observations. This new classification has been published in NASA Reference Publication #1217, in March 1989.

Extensive empirical testing of AutoClass has been valuable both from an applications perspective and from a desire to continue theoretical improvement. Over the last few years we and a variety of collaborators have applied it to several NASA and external databases. For example, a group at the National Library of Medicine (NLM), led by Larry Hunter, applied AutoClass to a large sample of proteins and discovered many interesting spatial patterns. These patterns included many that were previously known, and some that were new. In this application, the researchers had to carefully select a representative set of protein fragments and characterize them using the angles of the protein backbone. AutoClass then classified these angle sets into combinations that occurred with significantly high probability to be judged as a new class. The results of this analysis were recently published by NLM. This research showed a particular limitation in the current AutoClass, namely that although an angle of 3 degrees is very "close" to an angle of 357 degrees, these angles are very different numerically, and so AutoClass created unnecessary classes. We developed a method to handle angular data by modeling the angular distribution through an angular equivalent to a normal distribution. This extension is currently being implemented.

Another application of AutoClass was to the IRAS point source catalogue. This is very different from the IRAS low resolution spectral catalogue, and consists of 4 measurements (in 4 broad spectral bands) for nearly half a million sources. We extracted only those sources with sufficient flux in all 4 bands to be above the noise limit and applied AutoClass to this data set. The resulting classes showed distinct galactic distributions, even though only the spectral characteristics were used in the classification. Of special interest was a class that shows a distinct galactic bulge population above the galactic plane.

We identified several significant deficiencies of the current AutoClass implementation, as a result of this testing. These deficiencies were a result of simplifying assumptions made to turn Bayesian theory into the AutoClass algorithm. The main limiting assumptions are that the attributes are independent within a class, that all attributes are relevant for classification and that the classes themselves are independent. During the past year, we extended the Bayesian theory to remove these limitations, and have implemented these extensions in an experimental version (AutoClass IV).

To remove the independent attributes limitation, we allow the real-valued variables to be correlated and interacting discrete variables to be grouped into sets. For the real-valued correlations we allow a range from uncorrelated to fully correlated depending on what the data dictate. To remove the problem of irrelevant variables and dependent classes, we introduced hierarchical classes. Classes high in the hierarchy share some variables, and those lower in the hierarchy are distinguished by separately estimating the remaining variables. Any variable that is completely irrelevant is estimated at the root of the hierarchy and so is common to all classes. The hierarchy automatically groups together classes that are "similar". These extensions have shown greatly improved classifications on test data, but have also shown that the problem of searching for the optimal classification within this greatly expanded search space is the major limitation. Section 3.2.2 below provides a complete description of our related work in probabilistic search of massive spaces.

Improved Implementations of Autoclass

During the past year, we collaborated with the Intelligent Systems Technology Branch of our Division in using AutoClass as a test system for speed-up on parallel processors. The algorithm lends itself naturally to the task, because each case in the database could potentially have its own processor in the finest scale parallelization. This potential speed-up through parallel processing was tested by Andy Goforth (FII) and Phillippe Collard (CalSpace) using a parallel ADA implementation on several parallel ADA testbeds. They were able to show a speed-up linear in the number of processors. This research shows that AutoClass could be dramatically faster than its current implementation by porting it to a parallel machine, such as the connection machine. A reimplementation of AutoClass for the connection machine is being attempted by Thinking Machines Corporation. Further improvement in AutoClass speed could be easily obtained by rewriting it in "C", or similar numerical language. However, any such re-implementation should wait until the extended version of AutoClass (IV) has been fully developed.

The Bayesian approach to model-based data interpretation has proved itself both in the work done within FIA and in research by many other groups. It is a very powerful way of understanding what is happening in data generated by complex systems. So far, the research has concentrated on exploratory data analysis, using tools such as AutoClass, but it also has great potential in applications where a large amount of prior domain knowledge is available. Not only can the Bayesian approach find the most probable model (explanation, theory, hypothesis etc.), but it can also say how certain that model is, indicate probable alternatives, enable optimal decision making, make predictions, etc. This approach makes many important NASA applications feasible. For example, a potential application currently being investigated is global climate modeling/prediction. Currently, there are many global climate models, but there is so much uncertainty in their predictions that they cannot be relied upon, especially for important policy decisions. Not only do they not agree with themselves, they do not even predict the past. The Bayesian approach has the potential to overcome this problem by making as accurate a prediction as is possible, but also saying how accurate the prediction is. Bayesian global climate modeling is a huge task, and currently we are only investigating its feasibility.

Global climate modeling is only one of many potential applications we are investigating. Another related problem is ground cover mapping using remote sensing data. The current methods were developed many years ago, and are amazingly primitive compared with what is possible using fast algorithms and powerful data analysis methods.

The Bayesian approach can also be applied to time series analysis. NASA has many applications that require analysis of time series in space science and in operations. For example, solar wind data, variable stars, X-ray bursters, etc., are time series data sets where the underlying process is poorly understood. In NASA operations, the analysis of maintenance records, shuttle main engine test data, etc., may show interesting patterns or trends. In 1991 we began investigation of possible models for time series analysis, including Fourier methods, modeling chaos, modeling of shocks (discontinuities in the data, such as occur in the solar wind), hidden Markov models, and other more specific models. In 1992 we expect to extend these temporal models and develop implementations of them. These temporal data analysis tools should be very useful for preliminary data exploration in the domains mentioned above, but in addition, the Bayesian approach can develop domain specific temporal models for cases where there is already some understanding of the underlying process. For example, analysis of the shuttle main engine test data would look for vibration patterns that correlate with eventual failure of the engine. This kind of domain specific model building requires close interaction with the domain expert.

Milestones

1991: - Choose appropriate search strategy and complete AutoClass IV implementation.
- Publish a journal article describing all of the AutoClass research to date.
- Complete theoretical development of time series capabilities.

1992: - Finish implementation of time-ordered data classification and demonstrate applicability on an appropriate NASA application (e.g. STS engine test data or solar wind data)
- Distribute a stable version of AutoClass IV to interested collaborators.
- Investigate the application of Bayesian methods to climate prediction using remotely sensed data.

1993: - Demonstrate applicability of Bayesian methods to image reconstruction problems.

1994: - Demonstrate use of AutoClass for STS launch/landing site weather prediction.

3.2.2 Probabilistic Search

Peter Cheeseman, Wray Buntine, Bob Kanefsky, Will Taylor

Combinatoric search of massive spaces is important to a wide class of NASA-relevant constraint satisfaction problems. This research, initiated in 1991, investigates a new search method for combinatorially large spaces where traditional methods are overwhelmed, particularly in the class of hard problems called NP-complete problems, such as graph coloring, Hamilton circuit, Travelling Salesman, etc.

Our method works by taking the description of the variables, their possible values, and constraints to be satisfied, and creating a new description where there are probabilities for each variable taking on each of its possible values. Initially, these probabilities are all equal. Next, a function is constructed from the constraint descriptions that represents the expected number of constraint violations (as a function of the probabilities)--a type of cost function. This cost function is usually a simple polynomial in the probabilities and can be symbolically differentiated. The method employs a gradient descent where the probabilities are adjusted a small increment at a time to find the minimum expected number of constraint violations. If there are solutions to the problem, the probabilities will converge on one. Note that this approach does not require the invention of heuristics--they are automatically generated from the problem description. Also, unlike previous discrete approaches, such as backtrack search, the decision on how to adjust the probabilities is a global one, affecting all variables, and not just a local decision. Discrete methods seem to get trapped in local minima because they make local decisions.

A major complication with the above approach is that the probabilities must always be positive, so the problem is one of constrained optimization. We are investigating a way around this problem using a "penalty function" that blows up whenever the probabilities attempt to go negative. The simplest cost function is the entropy of the probability distribution, and using it leads to mathematics that is formally identical to that in statistical mechanics! We can prove that there are no local minima in the continuous search space unless the probabilities approach their extreme values of 0 or 1. We have not been able to prove that this approach necessarily finds a solution if one exists, but empirical investigations have so far been very promising.

A by-product of this research was a discovery of where the really hard problem instances occur for NP-complete problems. We found that they occur at a phase boundary between underconstrained and overconstrained problems. A paper summarizing this discovery has been written and submitted to IJCAI-91, but further investigations should lead to an extended paper for journal publication in 1991.

The major outstanding questions that will dominate research for the rest of 1991 and for 1992 include how the different cost functions that can be constructed affect the probability of the algorithm finding a solution if one is present, and how the approach can be extended to include NP-complete problems with global constraints. For example the problem called Graph-Partition requires finding a split of an even number of nodes into two equal sized sets that have the minimum number of edges connecting the two sets. Graph partition problems arise in circuit layout, for example. The simple cost functions that worked well for graph coloring break down for graph partition because the even split constraint is a global constraint on all the variables, and not a binary constraint between pairs of variables as in graph coloring. A statistical approach based on the central limit theorem is being investigated to get around this problem.

Milestones

1991: - Investigate the effects of different cost functions.
- Extensive empirical investigation of algorithm's effectiveness.
- Further theoretical investigations on strengths and limitations of approach.

1992: - Apply new search algorithm to realistic problems, particularly scheduling and AutoClass multidimensional search.
- Further theoretical and empirical investigation into the limits of this new approach.

3.2.3 Scientific Model-Building Assistant

Richard Keller, Michal Rimon
Michael Sims; Ames Intelligent Systems Technology Branch
Chris McKay; Ames Solar System Exploration Branch

Model-building is an integral part of all scientific enterprises. Scientists studying a particular phenomenon develop theories in order to account for novel observations and to make predictions about expected behavior. To validate their theories, scientists conduct *in vivo* experiments whenever possible. When it is impossible to carry out such direct experiments due to cost or other limiting factors, scientists build software models of the system under study (e.g., the human brain, a planetary atmosphere, a forest ecosystem) and then test their theories against those models.

Unfortunately, scientific model-building can be a time-intensive and painstaking process, involving the development of very large, complex computer programs. Despite the effort involved, scientific models cannot easily be distributed and shared with other scientists. This is because the implemented scientific models are low-level, idiosyncratic, and difficult for anyone but the original scientist/programmer to understand. In particular, the high-level structure and content of the scientific model is not obvious from the software implementation that confronts the would-be model user. Fortran and other general-purpose programming languages do not include the scientific terms and concepts that are natural to the scientist. Furthermore, important modeling and data assumptions are typically implicit in the low-level code that implements the model. These implicit assumptions cannot be easily inspected or modified by a potential new user. This is a significant deterrent to using another scientist's model; the appropriateness of assumptions is frequently the source of scientific debate.

To address these problems, we are constructing a *scientific modeling software tool* that serves as an aid to the scientist in developing, using, and sharing models. To facilitate model construction, the tool includes an interactive intelligent graphical interface, a high-level domain-specific modeling language, a library of physics equations and experimental datasets, and a suite of data display facilities. In constructing this tool, we are using a variety of advanced software techniques, including AI-based knowledge representation, automatic programming and truth maintenance techniques, as well as techniques from object-oriented programming, graphical interfaces, and visualization. Rather than construct models using a conventional programming language, scientists will use our graphical interface to "program" visually using a more natural high-level data flow modeling language. The terms in this language involve domain concepts (e.g., quantities, equations, and datasets) that are familiar to the scientist.

As a testbed for this research, we have developed a software prototype in the domain of planetary atmospheric modeling. This prototype is being used to reconstruct a model of the thermal and radiative structure of Titan's atmosphere that was originally developed in Fortran by C.P. McKay (ARC Solar System Exploration Branch). Although the current prototype is tied to the planetary atmospheres modeling domain, we are also investigating the applicability of these ideas to ecological modeling, where we are studying a model of the carbon, water, and nitrogen cycles in a forest ecosystem. This particular model was developed by S.W. Running (Univ. of Montana), and is in use by our collaborators in the ARC Eco-systems Branch. By studying two different application domains, we hope to gain insight that will ultimately enable us to develop a general-purpose scientific modeling "shell". The shell would be instantiated for a particular modeling task by supplying domain-specific knowledge.

We began 1991 by evaluating the prototype tool we developed in KEE on the Symbolics workstation during 1990. We identified several shortcomings in our prototype, and are now in the process of redesigning and reimplementing this system. Our redesign effort focuses on both the graphical interface and the knowledge representation structures that store the system's domain knowledge. We have tranferred our development efforts to CommonLisp on a Sun SparcStation to enhance portability. As of this writing, we have redesigned and implemented our representation for scientific equations. During the balance of 1991, we will design and implement representations for physical quantities, physical units, modeling assumptions, datasets, and various atmospheric domain objects. In addition, we will implement a mechanism for binding scientific equations -- an essential function in the model-building process. We will also reimplement the graphical user interface using a MacDraw-style interface that permits users to flexibly construct and modify scientific models by manipulating data flow graphs.

During 1991, we continued our collaboration in planetary atmospheric sciences with C.P. McKay. To ensure the applicability of our approach, we plan to extend the coverage of our modeling tool to a larger portion of McKay's original Titan model. For this purpose, we selected the planetary haze computation portion of the model and built a preliminary data flow graph covering this portion. We will implement the equations for planetary haze in our new equation representation. In parallel with this work, we began a collaboration with ecosystem scientists at Ames. We have studied portions of the Forest-BGC ecosystem model, and have developed data flow graphs for two major subportions of the model. By the end of 1991, we will complete our analysis of the entire model, in preparation for implementation during 1992.

A major long term goal of this work is to create a tool that will be directly usable by planetary scientists in their daily research and modeling activities. In particular, we hope to provide a facility for use in supporting the upcoming Cassini mission to the Saturn system. As a step in this direction, our thrust in1992 will be on completing and delivering a base-level working prototype to modellers in planetary atmospheric sciences. (We will also attempt to deliver a prototype version to ecosystem scientists, but our success in this endeavor depends on how much commonality in modeling exists across these disciplines.) In addition, we will begin development of advanced modeling facilities, including a mechanism for recording and maintaining scientific modeling assumptions. During 1992, we will report to the scientific community on our progress by writing a AI applications conference paper on our system's design and development.

Milestones

1991: - Demonstrate complete initial prototype for Titan atmosphere modeling.
- Extend coverage of Titan atmospheric model to planetary haze computation.
- Develop data flow representation for forest ecosystem model.

1992: - Complete "production" version of atmospheric modeling tool and deliver to planetary scientists for experimental use.
- Design representation for modeling of assumptions and a mechanism for maintaining assumptions.

1993: - Complete empirical evaluation and publish results of atmospheric modeling tool use.
- Demonstrate first complete implementation of eco-system modeling tool.

3.2.4 Model-Based Learning

Wray Buntine

The goal of this research is to develop generic tools for learning from data and from partial models of the domain, and to develop the capability to rapidly develop and tailor these learning tools for particular domains given, for instance, specification of the kinds of models that are of interest. The tools will be used to discover classifiers, which are prescriptions for predicting target variables in the domain (supervised learning or induction), or to discover models describing the domain (model learning). The input to the tools would be data and prior knowledge in the form of partial models of the domain supplied by the domain expert. The partial models can be as little as recommendations like "decision trees should be good classifiers for this domain," "there is no causal link between these sets of variables," or "this intermediate variable is important for classification." Alternatively, partial models may specify more intricate causal models and equational families describing connections between variables. The tools are being developed for the case when data is not abundant so the technique of "minimising empirical risk" cannot be applied. In the limited data case, Bayesian statistics and decision theory are the appropriate theory to use in designing the learning algorithms. Bayesian methods are the only methods from AI, statistics, information theory or neural networks that are specifically targeted at the smaller sample case. The use of these theoretical methods is important because empirical, *ad hoc* development of algorithms in the past has been time consuming and is often plagued by unexplained problems. Empirical validation of the algorithms is also important to check approximations made in interpreting the Bayesian theory. We do this empirical validation by applying the algorithms to a battery of recognised learning problems taken from the literature, or manufactured problems.

Research in 1991 consisted of extending and robustifying the work of Buntine's 1989 Ph.D. thesis on learning classification trees from data. The goal here was to validate the general approach (model-based Bayesian learning) on the supervised learning task of learning classification trees. Large scale comparative experiments were made comparing the Bayesian methods advocated here with techniques from classical statistics, AI, and information theory. The Bayesian methods were superior as predicted by the theory. The suite of tools developed for these experiments is currently being packaged for distribution to the research community.

Research in 1992 will consist of further extending and robustifying the tree learning system being developed, and extending this system to learn Bayesian networks. Initial research suggests the technique for learning Bayesian networks technique will be successful. Preliminary investigations are underway regarding use of these algorithms on some NASA applications in earth sciences, and other suitable applications are actively being sought. Also, techniques are being developed for several key problems in learning, such as incremental and large batch algorithms, techniques for handling missing values, and techniques for making multivariate tests in classification tree and classification rule systems. Pending perceived need, these will be incorporated into the current system.

Milestones

1991:
- Distribute system for learning classification trees as a research tool.
- Begin extension of tree system for learning Bayesian networks and probabilistic rules.
- Write journal article detailing theory for classification trees.
- Begin work on particular NASA applications of the learning tools, specifically in earth and space sciences.

1992:
- Begin research on function finding (regression) algorithms for learning functions from data.
- Incorporate techniques for specifying and improving partial models into the learning system under development.
- Distribute extended system for learning Bayesian networks as a research tool.
- Continue work on NASA applications of the learning tools.

1993:
- Update and maintain distributed code to improve functionality according to requests from users.
- Write journal articles reporting results as appropriate.
- Develop generic methods for rapidly generating learning algorithms from user specification of model families.
- Continue work on NASA applications of the learning tools.

3.2.5 Concept Formation in Classification and Planning

Douglas Fisher; Vanderbilt University

This research explores extensions to Fisher's earlier work on Cobweb, a system that organizes probabilistic concepts in a hierarchy, retrieves appropriate concepts by sorting observations through that hierarchy, and acquires knowledge by modifying the structure of the hierarchy. At each level of the hierarchy, Cobweb uses an evaluation function based on information theory to select the node that best matches an instance, incorporates the instance into that node's probabilistic description, and recurs to the next level. The approach is inherently incremental in that it interleaves learning with the classification process. At any stage, Cobweb can make predictions about missing features of the objects it observes, and experiments have shown that -- for many classification tasks -- the system rapidly converges on predictive concept descriptions.

In previous work, Fisher augmented Cobweb's control strategy to handle noisy domains, adapting statistical learning techniques to identify the optimal level in the concept hierarchy for prediction accuracy. Experimental results show that this approach significantly improves predictive ability in noisy domains. A more radical extension, started in 1991 and continuing in 1992, will extend Cobweb to form directed acyclic graphs instead of trees. This lets the system capture orthogonal classes in a concept hierarchy (e.g., mammals, birds, and reptiles vs. carnivores, omnivores, and herbivores). Such redundancy is important when the training data contain many missing attributes and when known attributes support classification only along certain dimensions. Potential applications include diagnosis of physical systems, such as those used in controlling the space station and planetary habitats.

In continuing work, Fisher is also adapting ideas from Cobweb to the management of search control knowledge. The basic approach involves carrying out concept formation over explanations and problem-solving traces. Problem solving becomes a matter of classifying a new problem and predicting parts of the solution trace as one descends the concept hierarchy. Noise-tolerant methods identify nodes at which one should abandon classification problem solving and complete the solution using domain knowledge. Preliminary experiments revealed an important and intuitive point that has not been identified in the literature on explanation- based learning: that use of learned rules should depend on the current status of problem solving (e.g., diagnosis of an aircraft component should be guided by earlier diagnosis of antecedent components). Early tests of this approach have focused on solving algebra problems, but Fisher is adapting the method to control search through fault trees for diagnostic reasoning.

In new work started in 1991 and continuing in 1992, Fisher is applying a similar strategy to the organization of plan knowledge. The emphasis here is on generating abstract plans based on classes of operators that have been grouped together in Cobweb's concept hierarchy. Such plans are desirable in uncertain domains where conditional actions are required, and Fisher's approach suggests a natural way to represent and acquire knowledge that supports the generation of abstract plans. His approach also shows how one can store the results of successful plans as macro-operators in the same long-term memory, and he has used related methods to refine descriptions of the operators used to generate plans. Such capabilities are essential for intelligent agents that operate in novel domains, as would occur in space construction or planetary exploration.

Milestones:

1991: - Develop a version of Cobweb that organizes its conceptual knowledge using directed acyclic graphs and carry out systematic experiments to evaluate its behavior in both natural and artificial domains.
- Extend the current work on concept formation over explanations, which relies on logical representations, to employ the probabilistic representations pioneered in Cobweb.

1992: - Initiate systematic experiments with the planning system to show its planning ability improves with experience.
- Continue experiments with concept formation over explanations, adapting this to reasoning with fault trees for diagnosis.

1993: - Continue experiments with planning system, investigating the effect of noise (unanticipated operator effects) on planning efficiency and the appropriate level of abstraction.

1994: - Extend Cobweb-based reasoning system to a large-scale diagnostic fault tree, and begin evaluating the system on actual telemetry data from this domain.

1995: - Incorporate Cobweb-based reasoning methods into a prototype diagnostic program for a subsystem of the space shuttle; begin initial field tests.

3.2.6 Efficient Learning Algorithms

Phil Laird

A major challenge for the machine learning field is to develop effective and reliable techniques for writing programs that learn (i.e., adapt and/or improve) over the long time periods that such programs will be used. Like any engineering technology, machine learning can progress only so far unless the fundamental theory is in place to enable us to understand the limitations of our methods. The purpose of the *Efficient Learning Algorithms* project is to apply existing theory to the development of practical algorithms and to develop new mathematics that enable us to design reliable and useful learning algorithms.

This research continues with a two-sided attack. On the one hand, we are developing, implementing, and prototyping learning algorithms with the purpose of determining their robustness and usefulness for NASA application. Currently we are studying the CDFI algorithm (developed and published last year) as a potential monitoring and diagnosis tool. A preliminary implementation in 1989 showed that it is practical in terms of real-time performance. The next step is to show that it is useful, and to this end we are combining the CDFI algorithm with other methods from the control theory literature. Beginning in March of 1991, we plan to begin a vigorous implementation of a proof-of-concept test of these ideas.

The second side of the research of the project is the theoretical study of learning methods, as a longer-term investment in new learning techniques. Currently the focus of this work is "speedup learning." The task for speedup learning is for a (presumably) correct program to modify itself and thereby become more efficient at executing the family of problems it has to solve, but simultaneously to preserve its correctness so that the learning process cannot introduce new errors. In the past year, we conducted and published a theoretical study of the applications of explanation-based learning (EBL) methods to this problem. The outcome of this work was a proof that standard EBL methods are too weak to be of much use for this particular problem. But, an interesting by-product of this mathematics was to show that there are new techniques that are more powerful than EBL in the context of speedup learning. Work is underway now to develop these new techniques and to implement them as programs. This research is progressing well; the first results should be available during late 1991 or early 1992.

Using the published CDFI algorithm as a basis, we will design and implement a "plug-in monitor" (PLUM) prototype. The purpose of this system is to observe any of a wide variety of devices, learn the normal operation characteristics of this device (with very little built-in knowledge about the actual device), and then to watch the device during its operation and detect sudden changes due to malfunctions or gradual changes due to drift. Various devices will be simulated in order to provide a test environment that serves to evaluate the practicality of the method. If successful, a software prototype will be released.

A new algorithm for learning a variety of program execution characteristics will be available for evaluation by the beginning of 1992. We plan to explore the potential applications of this algorithm to automatic program optimization. Two main applications are currently targeted: (1) expert systems that spend most of their time searching for applicable rules, and (2) planning systems that need to anticipate future inputs and to project possible reactions. Other applications are also likely, and several interesting theoretical issues will be investigated.

Milestones:

1991: - Proof-of-concept implementation for the CDFI Learning algorithm to help predict alarm conditions for scientific experimentation.
1992: - Preliminary report on new speedup-learning methods.

4 Autonomous Control

4.1 DTA-GC Instrument Control System

David E. Thompson, Richard Levinson, Deepak Kulkarni, Peter Robinson
Rocco Mancinelli, Lisa White; Ames Code SSS

This project is developing AI-based software that autonomously controls a new geochemistry laboratory instrument and that provides assistance in data analysis of planetary soils. This instrument combines two well-known materials analysis methods: differential thermal analysis (DTA) and gas chromatography (GC). Currently manufactured DTAs require a great deal of human supervision and decision-making during use. The software being developed will enable the DTA-GC to operate autonomously. This will relieve the laboratory staff's need to constantly attend the DTA and activate the GC. The development of this software will eventually culminate in a DTA-GC system that will operate autonomously in remote or hostile environments such as Death Valley, Antarctic Dry Valleys, or Mars. Although the coupling of these analysis systems will result in a more detailed characterization of mineral and soil samples, their combined use requires the development of new expertise, detailing how the data or results of the two types of analyses interrelate. Therefore, the software being developed provides a framework that acquires and stores data in a format which allows for testing of the incoming data from an experiment against computer-generated hypotheses about what minerals the soil sample contains. Such a framework greatly expedites the acquisition of a comprehensive mineralogic database that shows how thermal characteristics and mineral phase changes in the soil relate to chemical composition and mineral structure. The AI control software has the capability to recognize "interesting" events, or to anticipate particular features that will arise in the data, and re-program the oven heating function in real-time. This recognition requires the comparison of actual events with expectations which are generated before or during an experimental run. The selection of an efficient recognition methodology and an appropriate characterization of an event utilizes techniques in feature recognition, symbolic classification, Bayes net probabilistic associations based on geochemical models, and machine learning and discovery approaches which incorporate qualitative physics.

Significant work has been accomplished in both the geochemical and software development aspects of this project. The geochemists have performed about 140 sample runs on clays, sands, clay-sand aggregates, carbonates, salts, carbonate-salt mixtures, combinations of these, and mixtures of these sets with organic compounds. Several sets of standard reference samples were run to learn the peculiarities of the instrument and to standardize sample preparation techniques. Also, testing and calibration of both the DTA and the particular GC column for this coupled system has been completed, yielding reproducible results. Additional work includes increasing the sensitivity of the GC and decreasing the dead volume between the DTA and the GC by redesigning and fabricating a new interface for more efficient analysis. Also we must increase the range of compounds that we can analyze on the GC by adding another column and detector to the sampler. Columns are calibrated for certain gases, and some mixtures will swamp the signals from any co-occurring gases unless they are separated to a different column. Hence, one might, for example, dedicate one column to quickly-eluting gases, and one to slower flowing gases, so that data from each arrives in comparable time.

On the software development side, a full conceptual model for the "Control and Response" architecture has been created and implemented. This system allows copy-and-edit of experiment profiles and plan monitoring of new hypotheses and control protocol as these are suggested from the "Interpretation" component of the software. This Interpretation component consists of four modules: (1) feature identification from sensor signals, (2) classification and matching (or partial matching) of these features against library feature signatures, (3) explanation of the matches based on geochemical knowledge and environmental constraints, and (4) determination of control strategies. Although, not all of the interpretation components have been completed, the basic architecture is in place, and the feature recognizer and classifier are currently operational on real data. We have begun performance testing in comparison to our human geochemists; on a recent unknown mineral sample from the Antarctic, the Intepretation component was able to outperform the expert by enumerating several composition hypotheses, where the expert found only one. The geochemical explanation model, which fuses environmental information with the current experiment's data, is still in development. The bulk of the work to be done is filling out the particular chemical process relations involved, creating particular qualitative addition schemas for dealing with mixtures, and tying these to relevant condition statuses for the plan monitor system of module (4).

The Control and Response component represents the executive of the system. The main part of this element, the experiment design and control module, designs an experiment run as a series of commands for the hardware (notably DTA heating functions and GC sampling strategies). These programs are created by copying a selected previous program (guided by environmental status information) and then modifying that command sequence to suit the new situation. Execute, modify, and halt commands are sent to the hardware, based on the program's selection of best strategies, which are in turn grounded in incoming data and geochemical reasoning about the current status of the system and soil sample. The integration of the Control and Response component with the Interpretation modules is carried out through iteration of information between these systems. The Interpreter informs the Responder about its best explanation of what minerals or mineral families are contained in the soil sample under analysis, based on analysis of all data in at the time. The Responder then informs the Interpreter about execution plan assumptions while it reasons about possible futures implied by the information from the Interpreter's "best guess." As an execution deadline approaches, which is required due to a need to disambiguate between competing hypotheses of the chemical content of the soil in the oven, a best plan is selected and execution commands are sent to the hardware. Meanwhile, the Interpreter refines its explanation of the data and informs the Responder once more of changes to the external status of the Responder's execution plan assumptions. That information, ideally, impacts the heuristic monitors, which are directing the amount of planning time given to different strategies. This is possible because the explainer module passes on the Bayesian belief values associated with particular mineral assemblages directly to the Responder.

During this year, a major effort is being included to create specific data structures in the DTA-GC library and links in the Bayes net part of the classifier which represent both naturally occurring soil mixtures of minerals as well as hypothetical mixtures which might be expected to represent particular geologic environments on other planetary surfaces. This will allow our software to be able to reason both about situations in which minerals in the soil are independently undergoing their particular phase changes and decompositions as well as about situations in which the particular mineral assemblages in a sample interfere with each other during chemical and thermodynamic events, leading to masking or shifting of expected features in the data curves. Recognition of these situations are vitally important to developing an accurate qualitative reasoning module so that the possible justifications of the explanations make sense geochemically and are not just a simple superposition of a large sequence of partial match of information.

Milestones

1991: - Demonstrate autonomous data analysis and explanation functions.
- Completion of Responder and integration of s/w.
- Completion of environmental model.
- Completion of initial user interface for integrated s/w.
- Full integration with DTA-GC hardware.

1992: - Operational testing of complete system by ARC scientists.
- Enhanced system for data analysis assistance.
- Demonstration of generalized system including software for the XRD.

1993: - Further development of the analysis and control software to widen the range of instrument applicability.

4.2 Superfluid Helium On-Orbit Transfer (SHOOT) Flight Project

Eric Raymond, Jeff Shapiro, Gary Villere, Tim Castellano, Sharon Doubek, Frank Robinson

This project focuses on the development of software tools to support the monitoring and control of a Shuttle experiment called SHOOT: Superfluid Helium On-Orbit Transfer. Ames's role in this joint effort with the Goddard Space Flight Center is to provide all software used to control and monitor the experiment during all operational phases. Highlights of this effort include the first use of an expert system in space, an innovative use of laptop computer technology, a highly interactive, user friendly system used to teleoperate the payload from a remote ground station, and a simulation system used as both a development and training tool.

The SHOOT experiment is a Shuttle payload bay cryogenics experiment scheduled to fly on STS-54 in January of 1993 on the orbiter Endeavour. The SHOOT experiment consists of two insulated containers (dewars) of liquid helium connected by a transfer line. The helium is normally in a state called "superfluid" in which it appears to flow without viscosity. This superfluid component of helium has effectively no entropy and display quantum mechanic behavior at bulk, macroscopic level. The main experiment objective is to demonstrate the ability to transfer this odd liquid from dewar to dewar, much as would be done during a cryogenic servicing of a helium cooled orbiting telescope such as the planned Space Infrared Telescope Facility (SIRTF).

SHOOT will determine if all this is technically possible. Software being developed will provide facilities for the control and monitoring of the experiment as each hardware element and procedure are exercised. The software effort consists of three major components:

- AFDeX - An autonomous control, monitoring, and diagnostic system operated by the Shuttle crew to perform transfer operations.
- CMS - An interactive, user friendly Command and Monitoring System operated from the ground by a highly trained operator.
- CATS - A Command and Telemetry Simulator which emulates the payload for developmental testing of software and crew training.

AFDeX (pronounced /af-deck/) is a rule based expert system written in the CLIPS (a forward chaining production system language developed by NASA) and C languages. The system operates on a flight qualified GRiD 1530 80386-based laptop computer called the Payload and General Support Computer (PGSC) and will be located on the Shuttle's Aft Flight Deck (AFD). The system is designed to control the experiment without any support from ground or crew; it is completely "in the loop". The system interprets telemetry from the payload with respect to its current goals and sends commands to the payload in order to achieve these goals. If a problem occurs in the payload, AFDeX will attempt to diagnose the situation and recover. (Diagnosis is performed via empirical associations.) A user may optionally interact with the system to modify its behavior. The intent is to provide an autonomous control capability which is tolerant of a variety of failures and able to incorporate external knowledge from a user (perhaps as the result of earlier experiments in the mission) at execution time. Additionally this software provides the crew with real-time feedback from the payload which is used to coordinate a number of experiments involving thrusting of Reaction Control System (RCS) jets.

In contrast to AFDeX, the CMS is a ground based system for payload operation and control which provides an expert user with a highly configurable, fine grained interface with the experiment. While AFDeX contains expert knowledge tailored for specific phases of the experiment, the CMS is general purpose tool which is used in all phases of operation. This system along with the CATS simulator represent the current state-of-the-art in event oriented, mouse controlled user interfaces of the Macintosh computer.

Other responsibilities of the SHOOT software staff include supporting payload qualification testing, operation of the payload at Kennedy during post shipment checkout and integration, and operation of the payload from the Goddard Payload Operation and Control Center during flight. In addition, the SHOOT payload requires a final top off of helium about two days before launch while the experiment is integrated with the orbiter on the launch pad. The SHOOT software staff will assist in this operation.

Milestones:

1991: - Flight dewar tests. Flight-like operations under control of the CMS and AFDeX software.
1992: - Astronaut crew training begins with AFDeX and the CATS payload simulator.
1993: - SHOOT is launched on STS-54 on the new orbiter Endeavour.

4.3 Intelligent Interacting Agents

4.3.1 GEMPLAN/COLLAGE Multiagent Planning Systems

Amy Lansky, Andrew Philpot

This work focuses on the problem of generating plans for domains with multiple streams of activity that require complex forms of coordination. Thus, it deals with both action generation and action ordering, and must handle such issues as resource allocation and timing. GEMPLAN (and its successor, COLLAGE) is unique in that it provides a broader range of plan construction techniques than most planners, in its use of localized (partitioned) search to control the inherently explosive cost of planning, and in its integration of pre-planning and dynamic-planning.

One of our primary results of 1990 was to complete implementation, complexity analysis, and empirical testing of localized search. A localized domain description is one that decomposes domain activities and constraints into a set of regions. Such a description may be used to infer constraint localization semantics and, as a result, to enable the decomposition of the planning search space into a set of spaces, one for each domain region. Each search tree is concerned with the construction of a region plan that satisfies regional constraints. Shifts between search trees are guided by potential regional interactions as defined by the domain's structure. The localized search algorithm must also ensure that all search trees are consistent, which is especially difficult in the case of regional overlap. Benefits of localization include a smaller and cheaper overall search space as well as heuristic guidance in controlling search. Such benefits are critical if current planning technologies are to be scaled up to large, complex domains. Indeed, the use of domain localization and localized search are broadly applicable techniques that can be used by many kinds of domain reasoning systems, not just planners.

Work on the GEMPLAN and COLLAGE projects in 1991 may be broken down into four distinct subprojects. The first two projects deal with potential applications of both GEMPLAN and, eventually, the COLLAGE system (now under design and implementation -- see below). Work is necessary to augment the existing GEMPLAN system to accommodate these applications. The first application area we are exploring is aiding EOS scientists to help formulate plans for data analysis. Along with Rich Keller and a group of NASA Ames ecologists, we are formulating the concept of a Data Analysis Workbench, that provides both scientific modeling and data analysis planning assistance. While the goal of much ecological science is to build and verify models of earth processes and to use those models to project behavior, much of the day-to-day work done by these scientists (50% to 90%) is spent planning out how to do analysis; for example, deciding the best way to register data sets, which algorithms to use (e.g., for projection), which codes to use and on which platforms to run them, etc. Problems arise because much of the data analysis planning decisions are affected by multiple (sometimes conflicting) aspects of the projected data analysis task -- for example, the possible data input sources, the area of study, the type of data being studied and its properties, the underlying assumptions made by the various models that will be used, etc. Much of this decision making could be assisted (and some done completely) by automated tools. This kind of aid will be even more necessary given the anticipated quantity of data that will be made available by the EOS instruments, the wide spectrum of scientists that will be accessing that data, and the public's expectations of the results to be gained by Mission to Planet Earth. Tools that will assist the data analysis process and that will foster transfer and sharing of information and techniques between scientists will be quite important.

The second application area is facility construction. GEMPLAN/COLLAGE would take the role of a building or facility general contractor, constructing a plan of action that results in a finished artifice and takes into consideration all the requirements and constraints of the building, subcontractors, resources, etc. GEMPLAN has already been applied to a house building domain and we are currently working on a larger office building domain that will entail more flexible dynamic creation of regions. We anticipate that the construction application area will have potential in assisting with some of NASA's infrastructural requirements.

As stated above, one of our primary projects is the design and implementation of an entirely new GEMPLAN-based system. The new system, called COLLAGE (COordinated Localized aLgorithms for Action Generation and Execution), may be viewed as a generalization and improvement of the GEMPLAN architecture. COLLAGE will be a combined pre-planning/dynamic-planning system for multiagent coordinative reasoning, with flexible methods for metalevel control. It will generalize GEMPLAN's localized search mechanism to include much more flexible control over both internal regional search and external regional search. For example, it will provide broadened control over constraint activation so that plan construction can occur both before or during execution (i.e., constraints, whose application cause growth and modification of the emerging plan, will be applied both due to planning search as well as in reaction to the execution environment). Generalized external regional control will enable flexible transfer between and invocation of each of the regional trees, enabling, for example, parallel search of independent regions as well as other forms control over the order in which regions are searched. This will yield a true distributed reasoning architecture.

Efficiency is a key goal of the COLLAGE design process. We are designing the system so that the inevitable tradeoff decisions are justified experimentally. Among the technical target issues that we must ultimately address in COLLAGE include: (1) how to mediate between search and execution, especially in the context of a plan-construction search space (rather than a state-search space); (2) expanding and enhancing the constraint algorithms to improve coverage and efficiency; (3) enhancing user control over internal and external regional search via the use of a rule system and direct user interaction; and (4) investigating the use of run-time constraint application to handle some forms of disjunction (e.g., run-time choices) and looping (e.g., run-time expansion of loop bodies).

Our final subproject relates to developing a learning methodology for determining good domain localizations. Currently, the user specifies the decomposition of the domain that is used to partition the search space into a localized search space. Both empirical and analytical results about localized search indicate a tradeoff between increasing the amount of partitioning and increasing the amount of regional overlap. We hope to learn both domain specific and domain independent heuristics for determining good (and perhaps optimal) localizations. This work will be applicable to both GEMPLAN and COLLAGE, although initial experimentation is taking place with GEMPLAN.

Our plans for 1992 involve continuing along the paths of these four subprojects: development of applications and continued development of basic planning methodologies and algorithms in the context of COLLAGE, guided by the needs of these applications. By the end of 1992 we expect to have completed a version of COLLAGE that incorporates all of the existing mechanisms within GEMPLAN, plus, at least, rudimentary metalevel facilities for controlling aspects of internal regional search and external regional search. By this time, we also expect to have transferred most of our applications focus from GEMPLAN (in which we are currently testing applications ideas and implementing the localization learning techniques) to the COLLAGE testbed. The initial focus domain for COLLAGE will be facility construction.

In1992 we also expect to get publishable results about learning good domain localizations, although this learning facility will probably not yet be incorporated into COLLAGE. Finally, we hope to have constructed an interesting EOS domain application and to have fleshed out our concept of a data analysis workbench.

Milestones:

1991: - Identify and complete small demonstration within data analysis planning domain.
- Identify construction planning application within NASA.
- Complete first cut at localization learning algorithm.

1992: - Complete initial implementation of COLLAGE.
- Complete localization learning work.
- Develop architectural design for EOS data analysis workbench.

1993: - Demonstrate prototype EOS data analysis workbench.
- Experimentation with COLLAGE's flexible search control mechanism, especially within the context of selected applications. This will include work on parallel search of regional trees and mediation between search-based plan changes and reactively activatedplan changes.

1994: - Complete larger COLLAGE applications.

1995: - Experiment with integration of learning capabilities into COLLAGE.

4.3.2 The Entropy Reduction Engine Project: Integrating Planning, Scheduling, and Control

Mark Drummond, John Bresina, Rich Levinson, Andy Philips, Keith Swanson

The Entropy Reduction Engine project is a focus for research on planning and scheduling in the context of closed-loop plan execution. The eventual goal of the ERE project is a set of software tools for designing and deploying integrated planning and scheduling systems that are able to effectively control their environments. To produce such software tools, we are working towards a better theoretical understanding of planning and scheduling in terms of closed-loop plan execution. Our overall project has two important sub-goals: first, we are working to integrate planning and scheduling; second, we are studying plan execution as a control theory problem. The next two paragraphs consider these complimentary goals in more detail.

Traditional AI planning deals with the selection of actions that are relevant to achieving given goals. Various disciplines, principally Operations Research, and more recently AI, have been concerned with the scheduling of actions; that is, with sequencing actions in terms of metric time and metric resource constraints. Unfortunately, most of the work in scheduling remains theoretically and practically disconnected from planning. Consider: a scheduling system is given a set of actions and returns, if possible, a schedule composed of those actions in some specific order. If the scheduler cannot find a satisfactory schedule, then it will simply fail. The business of planning is to select actions that can solve a given problem, so what we need is an integrated planning and scheduling system to overcome the problems of scheduling alone. An integrated planning and scheduling system would be able to consider alternative sets of actions, unlike the stand-alone scheduler, which is unable to deviate from its given action set. We are working towards such an integrated system by incrementally constructing a unified theory of planning and scheduling that can be computationally expressed as practical software tools.

Most planning and scheduling work assumes that the job of the automatic system is done when a plan or schedule has been generated. For simplicity, we will use the term "plan" in what follows to refer to the output of an integrated planning and scheduling system. One of the first things that you learn about plans is that they are rarely ever perfectly predictive of what will happen. As Dwight D. Eisenhower observed, ``Plans are nothing, planning is everything". We agree with this view, since it tells us that the importance of planning does not lie in the existence of a single plan, but rather in a system's ability to re-plan, and predictively manage plan execution failures in light of continuous feedback from an environment. In the ERE project, we view plan execution as a problem in control; specifically, we formalize plan execution as a form of closed-loop control, where a plan describes a desired behavior, and feedback from the environment is used to measure deviations from this behavior.

During 1991, we have focussed primarily on the study of plan execution as a control theory problem. We have implemented a simulator to test our ideas and to enable rigorous empirical validation of our developing theory. Using this simulator, we have designed and tested a temporal projection system that is able to reason about the probability of goal satisfaction, given only prior knowledge of actions that can be taken and a statement of the goal to be satisfied. Using this system, we have undertaken the study of how plans can be used to increase the probability of goal satisfaction. We have discovered that for some problems, little or no plan is needed: the background probability of success is sufficiently high that no explicit "plan" is required. For other problems, plans are absolutely necessary to avoid high-probability failures.

One result of this year's work has been the design and implementation of a pair of algorithms, called Traverse and Robustify, to incrementally form plans which maximize the probability of goal satisfaction. After finding an initial "seed" plan, the algorithm improves the probability that that plan will satisfy the goal by finding high-probability plan failures and then fixing them, before they actually occur. This algorithm has been presented to AI and control theory audiences, and has helped establish critical connections between the techniques of traditional control and those of AI planning. We will explore and strengthen these connections in future work. Related ideas on the use of deadline information to decide when to plan and when to act were studied and will be connected to our incremental plan formation algorithm in future work.

As part of a project co-sponsored by DARPA and NASA, we organized a workshop in the spring of 1990 on benchmarks and metrics for the evaluation of integrated systems which served as an excellent and timely focus for discussion. Many people from academia, industry, and government attended, and some initial agreement on how to measure progress was achieved. A paper was written describing some of the lessons learned at this workshop. We have begun to classify problems, and plan on publishing a problem catalog to foster the empirical study of planning and control systems. As part of this work, we are producing a users' manual and set of programmers' notes for our simulator, which will be available to the research community in early 1991.

Our current simulation environment exhibits features that are common to many NASA missions, such as uncertain action outcome, the existence of exogenous events, and the requirement for goals that extend over intervals of time. Beyond this simulation environment, ERE technology has, over the past year, made an impact on the Differential Thermal Analyzer - Gas Chromatograph (DTA-GC) application (see Section 4.1 of this report), and we expect to have increasing interaction with this project in the future.

We have extended and to a certain extent implemented our ideas on the interaction between problem reduction and temporal projection. We are using the REAPPR system as our problem reduction engine; we have established an initial connection between REAPPR and other ERE components. Work on the use of problem reduction to provide search strategies to temporal projection has lead to interesting insights about the problem of learning reduction rules that serve to minimize the probability of backtracking by the projector.

Initial results in learning better causal theories have been obtained by Smadar Kedar in collaboration with members of the ERE group (see Section 4.3.4 of this report for more detail).

Perhaps most significantly, the ERE group has constructed a set of demonstrations to help explain our results in concrete terms. Our demonstrations cover the simulator, our probabilistic temporal projection mechanism, and the compilation of situated control rules (and the generality--specificity tradeoff contained therein, due to our work with Dr. Kedar).

In addition to developing our theory further, we plan to conduct several demonstration and evaluation projects. As part of the benchmarks and metrics effort, the ERE architecture will be evaluated on the Teleos hardware testbed during 1992. We also intend to carry out demonstration projects which show how our technology can affect NASA missions by improving safety, decreasing cost, and increasing productivity. In 1992, we will express our technology in terms that are grounded in specific NASA missions. In the past, we have studied a number of possible missions, and in the rest of 1991 we will focus on the following in more detail.

- Automated photometric observatory planning, scheduling, and control.
- Spacecraft command sequencing and execution.
- Orbiter processing (in conjunction with Zweben's KSC scheduling work, (described elsewhere in this report).

During 1992, we intend to select one of these applications and develop (or acquire) a simulation environment and scenarios which are of manageable complexity for research purposes, yet which preserve critical problem features of the selected application. The process of evaluating (and extending as needed) the ERE architecture based on these scenarios will begin in 1992.

The continuation of this application project beyond 1992 is intended to proceed according to the following plan. Once we have demonstrated adequate performance on the simplified scenarios, then we will scale-up the scenarios and re-evaluate system performance (and extend the system where necessary). Experiments will be carried out using the ERE application software to control the actual application hardware (in a safe test mode). Once confidence in system performance and robustness has been achieved, the application system will be deployed at the targeted site.

The primary topics on our research agenda for the rest of 1991 and for 1992 are the following:

- Managing metric time and resources;
- Planning to gain information;
- Designing complex behaviors;
- Learning problem reduction rules to improve quality of synthesized problem solving strategies;
- Affecting search control in response to real-time changes in the environment (in which the system is embedded).

We expect to publish research papers on these topics during 1992 and 1993. The milestones other than these research goals are outlined below.

Milestones

1991: - Completion and distribution of software and manual for our tileworld simulator.
- Completion of report describing set of benchmarks and metrics for planning tasks.
- Selection of an application problem (from above list).

1992: - Empirical evaluation of Traverse & Robustify algorithms.
- Evaluation of ERE architecture on Teleos hardware testbed.
- Development (or acquisition) of simulation environment and scenarios which preserve critical problem features of selected application.
- Demonstration and evaluation of ERE architecture performance on scenarios.

1993: - Organize a follow-up workshop on benchmarks and metrics.
- Publish results from hardware evaluation of system.
- Development of scaled-up scenarios and evaluation of ERE architecture performance on scaled-up scenarios.

1994: -Formalization and analysis of the interaction of problem reduction and temporal projection.
- Implementation of advanced user interface to system.
- Complete NASA/DARPA analysis of planning architectures.
- Experiment with using ERE application software to control application hardware.

1995: - Demonstratation of formal correspondence between temporal projection and truth maintenance reachability notions.
- Formalization and implementation of dependency-directed backtracking in temporal projection (and possibly in problem reduction, also).
- Deployment of ERE application system capable of integrated planning, plan execution monitoring, and reactive replanning in an operational NASA domain.

4.3.3 Acting to Gain Information

Stanley J. Rosenschein, Teleos Research

Teleos Research is carrying out a three-year program of research aimed at developing new techniques for constructing real-time systems that act to gain information. This is part of a larger research program focused on the design and implementation of intelligent reactive agents, which has had ongoing support from NASA and from agencies within the Department of Defense. Our work in this larger program has included the development of techniques for planning and acting in intelligent agents. This work has brought out the importance of perceptual action and inspired the new research direction of studying acting to gain information. This project seeks to extend previously developed techniques and to apply them to the problem of actively acquiring information from the environment in support of achieving particular goals.

We began by defining a collection of tasks that require information gathering in real-time. These tasks make use of both visual and tactile perception and include visual tracking of moving objects as well as searching for hidden objects by looking inside containers and by digging in sand. These task definitions are also of importance to the NASA/DARPA project on the development of benchmark tasks and evaluation metrics for intelligent agent architectures (described in Section 4.3.2 above).

The next step of the research program is to implement information-gathering strategies for these tasks using programming tools, including experimental tools developed previously as part of the research program. These implementations will be experimental and exploratory in nature, with the intention of revealing the patterns of reasoning and action that take place.

Based on the insights gained in the implementation of solutions to the target tasks, we will design new programming abstractions and tools that exploit regularities in reasoning and action to simplify the programming task. We expect to find that certain kinds of data structures for the representation of partial information will be applicable to a variety of situations and can serve as a useful generic data abstraction. We will implement these programming techniques, then use them to create new, more coherent, solutions to the target tasks. This cycle will be iterated again, refining the programming tools based on the experience of using them, and applying them to a newly-defined set of more difficult problems.

In addition, we will study an alternative approach to the construction of intelligent agents. Rather than simplifying the programming process for humans, we can move some of the burden to the agent itself, requiring it to learn appropriate strategies for acting to gain information in its environment. We will extend existing techniques for strategy learning to the more difficult case of learning strategies that affect the internal state of the agent (by gaining information) rather than simply affecting the external state of the world.

In previous work, we have studied the use of reinforcement-learning techniques, both in the context of a simulated world and in a real robotic system. Reinforcement learning is a kind of trial-and-error learning, in which an agent learns an action strategy by experimenting with different actions in an environment, which gives the agent feedback or reinforcement to tell it how well it is doing. It differs from explanation-based and speed-up learning, in that the agent is gaining information about the world rather than learning how to use more effectively the information it already has. It is also different from standard empirical concept learning because the agent is not told by the environment which actions it should take in which input situations; the agent must induce this by trying a variety of actions and determining which ones have the best expected outcome.

We will extend existing reinforcement-learning techniques to the more difficult case of learning strategies that affect the internal state of the agent (by gaining information) rather than simply affecting the external state of the world. In this case, there may be a very long series of actions that the agent must execute before it gets any reward from the environment (because it was able to gain information that allowed it to succeed in its task). Standard reinforcement-learning methods may not perform very well in such situations; if not, we will pursue the course of learning a model of the effects of the agent's actions, both on the external world and on its internal state, then dynamically and incrementally compiling that model into an effective action strategy.

Milestones

1991:
- Define a collection of target real-time information-gathering tasks.
- Implement information-gathering strategies for these tasks using currently available tools.
- Analyze patterns of reasoning used in implementing the strategies and define new abstractions, such as programming constructs and idioms, that exploit these patterns to simplify implementation.
- Build breadboard version of new programming abstractions.

1992:
- Re-implement the information-gathering strategies for the first task set using the new, extended set of abstractions and refine programming tools based on experience of applying them to the target tasks.
- Define a new set of target real-time information-gathering tasks that require more complex representational states and sophisticated perceptual action strategies.
- Investigate the use of learning techniques to allow an agent to learn perceptual action strategies from its experience in the world.

1993:
- Apply programming tools to complex task set.
- Test learning techniques on target tasks. Compare learned strategies to previously hand-coded ones.
- Integrate learning techniques with programming tools to allow the development of systems that are partially programmed.

4.3.4 Knowledge-Base Refinement in Integrated Systems

Smadar Kedar, John Bresina
Lisa Dent, Carnegie-Mellon University

The long-term objective of this research is to augment integrated planning and control systems with the ability to more autonomously refine their knowledge through learning from experience. Solving complex, real-time problems requires systems which integrate execution with planning and scheduling. To build such systems, a designer provides multiple knowledge sources to enable the system to perform multiple functions. One critical issue of integration is the detection and refinement of inconsistencies within and among the various hand-coded knowledge sources, in order to improve the system's performance according to various metrics (such as robustness, efficiency, etc.) Normally, extensive designer interaction is required to tune the system's knowledge throughout the system's lifespan. That tuning is time-intensive, and ad-hoc. One role for machine learning is to assist the system designer by autonomously and systematically detecting and correcting inconsistencies in the system's knowledge.

Our approach is to focus the research on one such state-of-the-art integrated system for planning, scheduling, and execution -- the Entropy Reduction Engine (ERE) discussed in Section 4.3.2 above. Our methodology is to develop various learning methods to address problems and inconsistencies discovered n the multiple knowledge sources as the ERE project attempts to integrate various components of this system.

Refining Operators and Domain Axioms: In building the ERE integrated system, knowledge of the possible actions in the world have been hand-coded into operator descriptions. Related knowledge has also been hand-coded as domain axioms which describe possible inconsistencies in the world model as a result of actions. Both operators and domain axioms are used in planning and execution. Through our experience in integrating both components we have discovered that when planning uses incorrect operator descriptions or domain axioms, invalid plans may result. And since the expectated outcomes of such plans may differ from the actual outcomes in execution, the agent may be unable to achieve its intended goal.

In 1991 we began developing a catalogue of learning methods which, given possibly incorrect operators and domain axioms, compare the results of both planning and execution to identify situations in which one of the knowledge sources is faulty, and the other knowledge source can be used to refine the first. One such method, based on Explanation-Based Learning (EBL) from failure, detects inconsistencies, finds possible causes, and refines the incorrect knowledge source in a general way based on those causes. One unique research contribution of this method is the ability to mutually refine multiple faulty knowledge sources. Previous work on EBL from failure has made the strong assumption that given one faulty knowledge source, a second complete/correct knowledge source is available to refine it. That is too strong of an assumption when modeling the real world. We've weakened that assumption so as to explore the possibility of refinement when all knowledge sources are faulty to some degree.

Learning and Generalizing Situated Control Rules (SCR's): SCR's are the pieces of compiled advice created by the ERE planner to be used by its reactive component during plan execution. In 1991 we applied EBL techniques to synthesizing SCR's with varying degrees of generality (a necessary component for ERE and for the above project). We plan to extend the generalization algorithm in novel ways to synthesize SCR's based on a richer set of goals. (Goals which mention time intervals, which intend to prevent as well as maintain situations, etc. need to be expressed and reasoned about in an integrated system for planning, scheduling, and execution.) We also plan to generalize a set of SCR's into one SCR which represents a repetitive strategy extracted from the SCR's, based on EBL Generalization-to-N techniques.

Refining Operators and Domain Axioms: We plan to continue developing a catalogue of methods for refining operators, domain axioms, and other knowledge sources within the ERE system. We plan to develop inductive as well as analytic learning methods for refinement. One proposed inductive refinement method would augment the operators with new non-deterministic outcomes when such outcomes are observed but not anticipated. In the course of time, if such outcomes are correlated with certain preconditions, the method will induce a causal relationship between preconditions and outcomes, and attempt to confirm causality by experimentation.

Applications: We plan to spend increasing effort in pursuing applications to "Debugging Assistants" for integrated systems, based on the methods described above. We expect to closely guide the development of the research by the needs of the application.

Milestones

1991: - Design and implement prototypes of the learning algorithms.
- Test the system in a simulated environment (the NASA Tileworld).
- Explore application to a NASA problem.

1992: - Continue development of basic learning algorithms, guided by a chosen NASA application.
- Design and test a prototype of the system in the NASA application.

1993: - Complete the full integration of adaptive learning methods into the ERE system.

4.3.5 Planning and Reactive Control
Nils Nilsson, Stanford University

This project (in its final year) focuses on reasoning systems that are situated in dynamic environments with other agents. These systems must react to the unexpected events caused by their inability to accurately model their own actions and the actions of the agents sharing their environment. This research addresses abstraction planning, planning formalisms, reactive programming languages, and efficient logical languages.

Three theses will be completed in 1991. The first concentrates on abstraction and planning. Jens Christiansen completed the implementation of PABLO which is a planning system that can learn or dynamically determine levels of abstraction during the planning process. This is advantageous for computational reasons -- details are delayed until more important decisions are worked out. Jens also co-developed a new formalism for characterizing non-linear planning systems. The truth criterion embodied by this formalism slightly extends Chapman's Modal Truth Criterion to allow variable instantiation constraints and conditional plans.

Also in 1991, Karen Meyers will complete her thesis on universal attachment. Universal attachment is a technique that allows logical proof systems to use algorithms to determine the interpretation of the syntactic elements of the language. Without attachments, only syntactic rules of inference such as resolution could be used to prove statements. Traditionally, logical systems have allowed semantic attachments to predicates where procedural code (usually in LISP) is called to determine the truth of a predicate. Karen has extended this technique to allow all elements of the language (e.g., constants and functions) to be interpreted by procedural code.

Professor Nilsson continues his work on reactive programming languages with the development of the ACTNET agent programming language. ACTNET programs have a circuit semantics where each program expands into a combinational circuit which is bounded by the depth of the circuit. Real-time behavior is guranteed because of the bound on the circuit depth. ACTNET is similar to the REX/GAPPS tools developed by the Teleos Corporation in that both are used to program real-time behavior within situated agents but, they differ in where they lie in the time-space tradeoff. REX/GAPPS programs compile much of the behavior into a large circuit while Nilsson's ACTNET allows circuitry to be synthesized dynamically. In 1991, Nilsson will continue the development of ACTNET and characterize what behaviors ACTNET can express. He will also contrast his technique with other behavioral programming languages like REX/GAPSS.

4.4 Integration of Symbolic and Numerical Control

4.4.1 Approximate Reasoning based Intelligent Control
Hamid R. Berenji

This research is focused on the development of techniques based on approximate reasoning and fuzzy logic control. We extend the conventional approaches of control theory by using the knowledge of a skilled human operator, in the development of an intelligent controller. A major focus of our research has been to provide learning capabilities for these controllers to automatically improve their performance by learning from experience. We have developed techniques for integrating approximate reasoning with reinforcement learning in the design of intelligent controllers. The approach selected here is to test this technique on a set of small but challenging control problems (e.g., cart-pole balancing, trajectory generation for a three-joint robotic arm, etc.) and extend the approach to larger size, challenging control problems (e.g., the rendezvous and docking operation of the space shuttle with a satellite in space).

In 1991, we extended our previous integrated approach for fuzzy logic control and reinforcement learning. The single neural elements used in our approach were extended to multi-layered neural networks. The new approach reduces the complexity of the older method by eliminating the need for the mathematical development of the trace functions and hence, provides a simpler method for developing fuzzy controllers which can fine-tune the membership functions used in their fuzzy control rules. We tested this approach on the cart-pole balancing problem and showed that it learns significantly faster than the other methods published in the literature. This approach will be tested using the proximity operations and docking simulation software that we have obtained from JSC.

We developed a method for using the Ordered Weighted Averaging (OWA) operators in fuzzy logic control and published the results in a joint publication with Professor Yager from Iona College. An OWA operator can model linguistic quantifiers such as *many*, and *most and* has the property of lying between the *and* function and the *or* function which are used in fuzzy logic reasoning systems.

Finally, we published the results of our joint work with Professor Whalen from Georgia State University. In this work, we developed an approach, which we called praxeologic logic, for reasoning with multiple intelligent agents who can consider the actions of other agents as evidence about the state of the world. This work combines concepts from the theory of evidence, epistemic logic, and utility theory.

Our plans center on three main topics: the continued development of the theory for our hybrid fuzzy logic control and reinforcement learning; the continued application of our control algorithms to the simulation of the rendezvous-docking operation and also trajectory control of a robotic arm; and the continued theoretical development of generalized aggregation operators (e.g., the Ordered Weighted Averaging (OWA) operators) in fuzzy logic control. We intend to provide a formal theory for our approach in integrating fuzzy logic control and reinforcement learning. This will provide a theoretical basis for addressing the stability and controllability problems of these controllers. The second topic involves testing our approach in a much larger problem such as the rendezvous and docking operation. We will perform this project jointly with Bob Lea's group at Johnson Space Center. Lastly, we will publish another joint paper with Professor Ronald Yager on the results of our investigations of the OWA operators.

Milestones:

1991: - Demonstrate a neural learning component for fuzzy control rules.
- Complete the application of our hybrid fuzzy logic control and reinforcement learning technique to the control of a 3-joint robotic arm.
- Develop and publish a general approach in integrating fuzzy logic control with reinforcement learning.

1992: - Demonstrate fuzzy logic control system using STS docking simulation software.
- Extend the theory for the generalized aggregation operators in fuzzy logic control.

1993: - Complete and publish an analysis of NASA domain applicability of fuzzy methods.

1994: - Demonstrate a complete integrated prototype of an operational fuzzy logic-based control system.

4.4.2 Fuzzy Control

Lotfi Zadeh; University of California at Berkeley

The major goal of this research is the development of computationally-effective methods for the analysis and design of intelligent control systems which operate in an environment of uncertainty and imprecision. In 1991, research was continued in two important application areas: (a) replacement of a skilled human operator by a fuzzy rule-based system; and (b) replacement of a domain expert by an FA-Prolog-based system which has the capability for representing facts and/or rules which are fuzzy and possibly probability qualified, with probabilities expressed as values of linguistic variables exemplified by *low, high, very low, not very high, etc.* The use of such probabilities provides an alternative method for dealing with dispositions, that is, with propositions which are preponderantly but not necessarily always true. Dispositional knowledge plays a central role in commonsense and non-monotonic reasoning, but in contrast to the techniques based on predicate logic, dispositional knowledge in fuzzy logic is assumed to be probabilistic in nature, with the understanding that the underlying probabilities are linguistic rather than numerical.

A mode of reasoning which plays an important role in our research is that of *interpolative reasoning*. This mode of reasoning is closely related to system identification, learning from examples and neural networks. Interpolative reasoning is particularly important in association with fuzzy rule-based systems, providing a way of computing the response to an input which does not match precisely the antecedent of any rule in the rule base. A new approach which is under development reduces the problem of interpolation in the case of sparse input-output data to the solution of a set of linear equations with fuzzy coefficients. As an alternative to the classical Lagrangian method, the solution of such equations is carried out numerically through the use of a version of FA-Prolog. In addition, the use of a conjunctive rather than a disjunctive combination of fuzzy if-then rules is under development for the case where there are gaps between the rule antecedents.

Our research has a direct bearing on the analysis and design of systems which may be employed directly or indirectly in space missions of various types. Experience with fuzzy-logic-based control systems has shown that the use of such systems enhances the robustness of system performance and reduces power requirements.

We plan to continue the development of the techniques described above and, in addition, investigate the following problem areas:

1. Develop methods of imprecise matching based on the use of possibility and necessity measures, and employ such methods for improved approximation in interpolative reasoning.

2. Apply neural network techniques to the adaptation of membership functions, based on the work of H. Berenji and C.C. Lee.

3. Develop methods of qualitative probabilistic reasoning using linguistic variables and FA-Prolog.

Milestones

1991: - Develop methods for interpolative reasoning for the case of sparse input-output data.
- Investigate the conjunctive processing of fuzzy if-then rules.

1992: - Develop an inference engine for uncertain knowledge employing the techniques of fuzzy relational equations.

1993: - Develop inference techniques for dispositional knowledge.

1994: - Extend FA-Prolog to decision-making with linguistic probabilities.

4.5 Machine Learning for System Maintenance and Improvement

4.5.1 Icarus: An Integrated Architecture for Learning

Pat Langley, John Allen, Wayne Iba, Kate McKusick, Kevin Thompson

The Icarus project is developing a framework for the control of autonomous intelligent agents. The goal is a single system that covers a broad range of cognitive behavior, including recognition of physical objects and events, generation of plans, and execution of motor skills. The eventual aim is to integrate these aspects of cognition into an agent that can interact with an unpredictable environment and acquire knowledge from its experience. Potential applications include NASA missions in which an autonomous agent might function in lieu of a human astronaut, as in planetary exploration or the execution of deep-space scientific experiments. In fact, such an agent would be suited to any hostile environment where it would be costly or hazardous to send humans: to do machine diagnosis/repair or substance cleanup in radioactive areas, or as a "trouble-shooting" rover in support of a martian or lunar habitat.

The framework assumes a single underlying organization for long-term memory -- a hierarchy of probabilistic concepts -- to store three types of knowledge: physical object concepts, plan knowledge, and motor schemas. Using a process of heuristic classification, the agent stores new experiences in the hierarchy, which effectively organizes knowledge in long-term memory. Intelligent action emerges not only from large amounts of domain knowledge, but also from the ability to efficiently find the "best" knowledge for a given goal and situation. The working hypotheses of Icarus are that a single long-term memory can represent many kinds of knowledge, that heuristic classification can retrieve this knowledge reliably, and that a single learning mechanism -- concept formation -- is sufficient to incrementally acquire knowledge from experience.

Icarus consists of three major components: Labyrinth, Daedalus, and Maeander. The Labyrinth system classifies complex physical objects and stores abstractions of them in long-term memory. The agent can use this stored knowledge to recognize similar objects and to guide its interaction with an object it may need to manipulate or avoid. The Daedalus system generates plans and acquires plan knowledge by classifying and storing components of successful plans in long-term memory. In drafting plans, the agent generates sequences of actions that it must follow in order to achieve its goals. By storing successful plan components in long-term memory, the agent learns appropriate responses to situations similar to those it has experienced. The Maeander system acquires motor schemas, which the agent can use to recognize a familiar action or to execute an action included in a plan. These three components are currently separate, but integration into a unified architecture is a research priority.

In 1991, we concentrated on testing these components in a variety of domains. Experiments with Labyrinth revealed effects of training order, which led us to develop an alternative method for incremental concept formation. We implemented these ideas in a system called Arachne, which carries out more restructuring of its long-term memory. Preliminary results suggest that this approach is more robust than the technique on which Labyrinth relies. Theoretical analysis of Labyrinth predicated that background knowledge should considerably aid the system in certain domains, and experiments with the system confirmed this predication. Studies with Daedalus showed that learning leads to reduced search, but also to increased retrieval time, suggesting the need for modified retrieval mechanisms. One approach we are pursuing involves the use of attention to focus only on a few relevant features; another involves a version of derivational analogy that can reuse entire stored solutions on similar problems. Experiments with Maeander also produced improved performance with experience, but suggested a sensitivity to certain learning parameters, which we will attempt to eliminate in future versions; the approach taken in Arachne may help on this issue.

The current year also saw initial work on integrating the three components into a single architecture, which we will continue in 1992. In particular, we will:

1. Extend Daedalus to store and retrieve complete problem-solving traces and abstractions of these structures, instead of the operator selection rules it currently employs. This will require replacing the current storage and retrieval mechanism with Labyrinth, since problem-solving traces can be viewed as complex structured objects. This will also require modifications to Labyrinth to support new types of tests.

2. Modify Daedalus to represent both states and operators as qualitative states, describing them in terms of the changes that occur while they are active. In this framework, the Meander component must also be revised to use the same representation for motor schemas, with these becoming simply very detailed plans. Execution would then involve 'running' a qualitative history that include primitive motor operators that the agent must execute for a specified length of time.

3. Interleave the processes of planning, execution, and perception. The central issue here involves when to halt Daedalus' deliberation in order to execute and/or sense. We plan to store

conditional probabilities on abstract problem-solving traces for use in these decisions. In cases where backtracking is unlikely, the agent could start to execute a plan before its complete specification. In situations where violated expectations are unlikely, perception could be bypassed.

4. Design mechanisms to handle interruptions and changes in goal priorities. In this scheme, classification of a new state may cause the retrieval of a stored problem that differs from the one currently being pursued. If the retrieved problem has higher priority than the currently active one, the latter will be suspended and the agent will pursue the more urgent one. Once this has been handled, control will pass back to the original problem, unless another one has taken over in the meantime.

We expect to make progress on each of these issues during 1992, although we expect to implement running versions of only the first two in this period. We have identified a simplified scenario in which to test these ideas. Shifting from our original focus on learning, we have decided to direct most of our energies toward working out the details of a unified representation for objects, events, plans, and motor schemas, since this must precede progress on the acquisition of such knowledge.

Milestones

1991: - Comple initial implementation and evaluation of Labyrinth, Daedalus, and Maeander systems.
- Modify Daedalus to employ Labyrinth in storing and retrieving plan knowledge.
- Extend Labyrinth to let it store and retrieve knowledge of plans, actions, and places.

1992: - Modify representation to store plans and actions in terms of executable qualitative states.
- Begin to interleave processes of planning, execution, and perception.
- Design mechanisms for handling interruptions and goal conflicts.

1993: - Complete initial integrated version of Icarus that interleaves planning, execution, and perception, and that can handle interruptions.
- Begin evaluation of Icarus in an autonomous agent testbed.

1994: - Complete evaluation of the Icarus architecture in autonomous agent testbed and begin adaptation to appropriate NASA intelligent agent task.

1995: - Incorporate Icarus software into a prototype inspection/diagnosis or maintenance device for STS or SSF.

4.5.2 Machine Learning and Planning in Dynamic Environments

Jaime G. Carbonell, Matthew Mason, Tom M. Mitchell; Carnegie Mellon University

This research effort addresses the design and implementation of autonomous agents that integrate aspects of sensing, planning, execution, and learning. The effort is composed of two related projects. One project focuses on developing a meta-reasoning architecture that lets one explicitly reason about different planning methods, experimentation strategies, and learning techniques. The other explores learning in the context of manipulation tasks that involve sensing, reacting, and planning.

Carbonell and his colleagues are extending Prodigy, an integrated architecture that builds on Steve Minton's earlier work with this system to include reasoning about meta-level actions. The resulting the system should be able to reason about when to plan vs. execute a canned procedure, when to modify a plan vs. replan, and when to experiment vs. apply existing knowledge. Upon making such decisions, Prodigy would use an explanation-based learning method to store a generalized rule that bypasses the need for future reasoning in similar cases. Such adaptive behavior will be essential to intelligent agents used to explore unfamiliar environments or that interact with complex physical situations.

Within this framework, Carbonell is developing methods for systematic experimentation that can be used to extend an incomplete domain theory. This approach notices when some plan fails to achieve a desired goal, and then attempts to determine the source of the problem by varying the nature of the objects involved or by varying parameters on the actions (e.g., the magnitude of an applied force). Experimentation may reveal that an existing operator description is incomplete, in which case the system adds some new condition or action to improve its domain theory (and thus it planning ability). The researchers have tested this system on some simple tasks, including production of optical-telescope mirrors. In 1992, he plans more extensive tests in a complex 60-operator domain based on a factory planning task. In addition, the system will be extended to support the representation and use of operators with probabilistic outcomes, and the ability to reason about alternative planning methods and about various learning methods.

In another project, Mitchell and Mason have used a hand-eye testbed to explore learning in manipulation tasks. The basic task involves tilting a tray so as to slide an object to a desired location. Despite its apparent simplicity, this task has proven challenging to the robotics community, making learning an attractive approach to acquiring the necessary domain knowledge. Mitchell and Mason have constructed a number of simple learning agents that operate in this testbed. Experiments have shown that these agents can learn effective manipulation strategies that produce 95% success rates at positioning objects at randomly selected target positions and orientations. Furthermore, such learning techniques are effective across a variety of irregular object shapes and in the presence of complicating factors, such as additional invisible objects in the tray. These results represent one of the first robotic tasks for which learning techniques significantly outperform state-of-the-art analytical planning methods.

In 1992, Mason and Mitchell plan to extend this approach to handle continuous representations of the physical world. To this end, they will use a goal-driven method for clustering states into predictive classes. They will also develop and evaluate new agents for the tray-tilting domain that employ different amounts of background knowledge in the learning process, testing them across a wider range of situations than in previous experiments. One promising approach would take advantage of a naive theory of physics to constrain hypotheses about objects' behaviors. Finally, the researchers will begin to extend the approach beyond tray tilting to the more complex tasks of pushing and grasping objects. Although relatively simple, these domains are relevant to NASA missions that may require robotic manipulation as a replacement or an adjunct to human operators.

Milestones

1991: - Conduct empirical evaluation of experimentation methods.
- Begin extension of Prodigy to support meta-level reasoning.
- Demonstrate experiments of increasing complexity in the hand-eye testbed.

1992: - Complete meta-level version of Prodigy architecture.
- Extend Prodigy to support operators with probabilistic outcomes.
- Extend hand-eye learning systems to handle continuous domains.
- Develop learning methods for manipulation that incorporate a naive physical theory to constrain hypotheses.

1993: - Run systematic experiments with extended Prodigy architecture.
- Run systematic experiments with extended hand-eye methods.
- Demonstrate pushing and grasping tasks in the hand-eye testbed.

1994: - Run systematic experiments with pushing and grasping tasks.
- Select the best learning system from previous studies with the hand-eye testbed and begin more detailed evaluation on complex manipulation tasks.

1995: - Complete evaluation of hand-eye learning system and begin incorporation into a teleoperated manipulation device, allowing a continuum from remote to autonomous control.

4.5.3 Learning to Use Devices

Paul Rosenbloom; USC/Information Sciences Institute

At the request of Rosenbloom's colleagues at Ames, this research effort changed significantly in 1991. Through 1990, Rosenbloom focused on basic improvements to the Soar architecture. Results during the previous year include restrictions on Soar's representation language that eliminated expensive chunks, extensions to the data chunking mechanism to support new types of knowledge-level learning , and demonstrations of the utility of combining analogical retrieval with background knowledge.

The new thrust of Rosenbloom's project involves reasoning about physical devices. These devices can be as complex as the shuttle's main engines, or as simple as a light switch. An intelligent agent situated in the world must be able to understand these devices and use them to achieve its goals. In particular, this project will model problem solving and learning in the domain of physical devices. Knowledge about a device will be represented in three basic forms: functional models that specify device behavior; structural models that specify device components and operations; and mappings between functional and structural models. Given a goal to accomplish (e.g., purge the lines in a fuel cell), problem-solving methods will be used to generate a sequence of actions to achieve that goal.

Learning occurs when the problem solver encounters an impasse, suggesting an omission in one of its models or its mapping. In such cases, the system will draw on multiple sources of knowledge to augment its understanding. These sources include instructions found in a manual, background knowledge about other devices, and active experimentation with the device. The system will construct or revise models of the device from these knowledge sources, which it can then use to achieve the original goal or similar ones.

The system will have two main components, one responsible for problem solving and the other for model acquisition and repair. Given an externally supplied task, the problem solver consults the model of the device to determine how to use it to achieve the task. The resulting plan is then executed on the device, and the observed result is compared to the model's prediction. If the model's prediction was wrong, or the problem cannot be solved at all, then the model is probably in error. In this case, the repair component consults the knowledge sources to find a repair that allows the problem to be solved. In this framework, problem solving and model acquisition are intertwined, rather than governed by some central executive.

Rosenbloom will cast his system within Soar, a cognitive architecture that he has already applied to many other domains in the past. As described in Section 4.5.5 below, Soar represents knowledge in terms of preference rules, which it uses to direct search through various problem spaces, and which it acquires through a learning mechanism called "chunking". In the domain of device use, chunking will let the system acquire mappings between different device models, as well as search-control knowledge to improve the use of devices.

Milestones

1991: - Select an initial NASA instrument or device to drive research.

1992: - Develop and evaluate an initial problem-solving system that can use correct functional, structural, and mapping models for physical devices.
- Run systematic experiments to evaluate the system with and without chunking.

1993: - Develop a simplified instruction manual and background knowledge sources for a physical device, along with methods for searching these knowledge bases.
- Implement methods for acquiring the mapping between functional and structural models from manual and background knowledge, assuming correct functional and structural models.

1994: - Implement methods for acquiring a structural model and its mapping to a behavioral model, assuming a correct functional device model.
- Implement methods for revising models from experimentation with a physical device.

1995: - Extend methods to simultaneously acquire functional models, structural models, and mappings between models from all sources of knowledge.
- Evaluate the system both as a machine learning system and as a simulation of human learning and performance.
- Begin to adapt the system for semi-autonomous control of a NASA device used on the Shuttle.

4.5.5 Soar and the External Enviroment

John E. Laird; University of Michigan

Soar is a general cognitive architecture that integrates problem solving and learning. The system represents knowledge as production rules that specify preferences for certain states, operators, or goals, and it uses this knowledge to direct search through various problem spaces. Before making a decision about which state to expand or operator to apply, Soar's rules add elaborations and preferences to working memory, which it then uses to select a state or operator. The architecture acquires knowledge through "chunking," a form of explanation-based learning that caches the results of previously solved problems.

Professor Laird's research at the University of Michigan uses Soar as a framework within which to develop an intelligent agent that interacts with an external environment. During 1990, Laird and his colleagues connected Soar to a Hero 2000 mobile robot, which has a variety of sensors and effectors. The resulting system, Hero-Soar, was able (in real time) to search its environment for cups, pick them up, and, after finding a box, drop the cups in the box. In another system, Robo-Soar, they have linked the architecture to a Puma robotic arm and an accompanying vision system. Laird has used this testbed to examine issues of reactive planning and execution, in which one is always reevaluating decisions in light of changes in the environment. If Robo-Soar has previously encountered the current situation, it responds immediately; otherwise, the system falls back on problem solving.

In 1991, Laird has been using these two testbeds to further develop a robust agent for interacting with external environments. For instance, if Soar is to interact with an environment, it must be tightly integrated with a perceptual system. One key idea is the use of visual attention as the mediator between low-level vision and high-level cognitive control. Visual attention is used as a method of limiting and localizing information within the visual field. The core of the model is a collection of operators that are sufficient for fixed-eye visual tasks in two dimensions. The model has been successfully used to account for psychological data on visual perception. Work in 1992 will focus on integrating these ideas into one of the robotic testbeds, for use in directing visual attention in these domains. Ultimately, this work could lead to improved vision systems for satellite reconnaissance, as well as for planetary exploration. The psychological aspects of the project could also lead to better interfaces for use in mission control.

In another effort, Laird is studying the important issue of interruption, in which an intelligent agent must delay work on a problem because a higher-priority problem unexpectedly occurs. This actually involves a number of issues, including detection of the alternative task, interruption of the ongoing task, possible interleaving of the two tasks, resumption of the interrupted activity, and learning from the experience. Work in 1992 will examine different forms of interaction between the interrupted and interrupting task, the effect of different encodings for task knowledge, and methods for remembering information about the interrupted task. This problem is relevant to NASA missions, since autonomous agents in construction and maintenance situations will have to juggle many tasks at the same time, such as the replacement of multiple ORU's (orbital replacement units), some critical and others not.

In 1990, Laird demonstrated Robo-Soar's ability to learn new skills from interacting with a human. During 1991, this work has focused on more autonomous methods. Here the system has basic capabilities for interacting with its environment, but it has no internal model of how its actions affect the world. Without such a model, it cannot plan its actions but instead must rely on reactive behavior. In order to acquire a model, Soar must perform actions in the environment, observe their effects, and induce the cause-effect relationships between them. Laird has implemented an initial system along these lines and has tested it for two simple domains, in which the relevant effects are close to the locus of actions in space and time, and in which the space of possible features is small. Work in 1992 will extend this approach to situations involving noise, feedback delays, and limited perception. This work has implications for planetary exploration, in that programmers cannot fully anticipate the effect of actions on an autonomous agent's environment.

Another related effort in 1992 will extend Soar to represent, use, and acquire knowledge of time-dependent behavior. Given a task, such as replacing an ORU, there may be multiple solutions that involve speed/accuracy tradeoffs that one must reason about. In other situations, there may be more goals than one can satisfy within the time available, and one must decide which one to abandon. Also, the activation of new goals may introduce new temporal constraints that suggest the interruption of the active problem. To support such reasoning, Soar will associate a `time for completion' with each operator, which it will learn from experience with actual executions of these operators. These estimates will be refined when predicted times diverge from the observed ones, using methods related to those for operator acquisition. This work is important to NASA missions in that many problems that arise in construction, maintenance, and exploration will have a time-critical nature, and an intelligent agent operating in these circumstances must be able to represent and reason about the times required for various actions.

The most important aspect of Laird's research program is its emphasis on *integration*. Each of the subprojects will be tested in the same domains, encouraging the development of a unified knowledge base that can be used to control an integrated intelligent agent. This is a long-term goal, but the issues of attention, interruption, operator acquisition, and time-dependent behavior all lie along the path toward this goal.

Milestones

1991: - Expand the model of visual attention to account for a wider range of psychological data.
- Extend the work on unsupervised learning from external environments to domains in which behavior is conditional on environmental features, to more complex actions, and to hierarchical actions.

1992: - Augment Soar to support interruption of ongoing tasks, decisions about relative task priority, and recovery upon returning to interrupted tasks.
- Augment Soar to include an internal model of time, and test the resulting system on tasks which involve alternative actions that vary in the time they take to execute and in their reliability.
- Implement the resulting methods in both Hero-Soar and Robo-Soar.

1993: - Adapt the attention model as basis for a Soar robotic vision system that can develop visual routines for new goal-directed tasks.
- Develop an integrated hand-eye system based on Soar, and begin evaluation of the system on manipulation tasks.

1994: - Complete evaluation of Soar hand-eye system and begin incorporation into a teleoperated manipulation device, allowing a continuum from remote to autonomous control.

1995: - Incorporate the Soar hand-eye system into an in-flight manipulator.

4.5.6 Revising Incomplete and Incorrect Domain Theories

Raymond Mooney; University of Texas, Austin

This project focuses on reasoning and learning in the presence of partial background knowledge. Most work on reasoning assumes that one can construct complete proofs that are grounded in logically sufficient facts. Similarly, most work on explanation-based learning relies on the presence of a complete domain theory and has no way to extend the domain knowledge they are given at the outset. Raymond Mooney's work at the University of Texas is attempting to move beyond these limitations.

In response to the first issue, during 1990 and 1991 Mooney developed an AI system for abductive reasoning which constructs explanations that require default assumptions. His Accel system uses a metric for explanatory coherence in the construction and evaluation of these abductive explanations. He has used several examples to show that the evaluation metric, which measures how well an explanation connects a set of observations, is superior to standard simplicity metrics at choosing the best explanation. Accel uses this metric as a heuristic for search during backward chaining to ensure the efficient construction of high-quality explanations.

Mooney has run experiments demonstrating that Accel's heuristic search method greatly reduces run time while still achieving optimal explanations in terms of the coherence metric. He tests to date have focused on examples that involve generating explanations for animal coloration and human intentional behavior. However, the work has clear applications to the diagnosis of complex physical devices, such as the control systems for the space shuttle. During 1992 he hopes to identify a NASA diagnostic domain and begin adapting his approach to abduction for this purpose.

With regard to learning from incomplete background knowledge, during 1990 Mooney developed IOU (Induction Over the Unexplained), an approach that combines existing inductive and explanation-based techniques. The system used explanation-based methods to learn part of a concept and uses inductive methods over unexplained aspects of examples to impose additional constraints on the final concept definition. Experiments in two financial domains (classifying stocks and predicting bankruptcy) demonstrated IOU's ability to use incomplete theories to learn more accurate concepts from fewer examples than a purely inductive method like Quinlan's ID3. He also used methods from learnability theory to formally prove that IOU can learn from fewer examples. Finally, he has used IOU to model psychological data demonstrating the effect of background theories on human concept acquisition.

During 1991, Mooney extended this approach to handle both overly general and overly specific background knowledge. The result was Either, a system that identified faulty parts of a domain theory through a process of abduction, using the Accel system as a subroutine. The new algorithm can correct background knowledge with both missing and extra inference rules, as well as rules with missing and extra conditions. In cases with unnecessary structure, Either simply deletes the offending components; in cases with missing structure, it invokes a decision-tree algorithm to induce a new rule or condition. The search for improved domain theories is directed by an evaluation function that measures ability to cover the training data.

Mooney has tested this approach on its ability to learn rules for recognizing promoters in DNA sequences, which initiate the transcription of genes. Experiments revealed that Either improved on an initial overly specific theory culled from the biological literature, but that its use of this theory as a starting point led to better results than a simple decision-tree algorithm. One limitation is that the current system handles only propositional domains; in 1992, he plans to extend Either to cover domains that involve relations among components, which will be necessary for many NASA applications. To this end, he may adapt Quinlan's FOIL algorithm which applies techniques similar to those used in ID3 to relational data. This approach should mesh well with the abduction module, which already handles first-order descriptions.

Milestones

1991: - Developed and tested Accel, an algorithm for constructing abductive explanations of observed facts.
- Extended approach to learning from incorrect background knowledge, producing and evaluating the Either system.

1992: - Refine the heuristic search procedure used in Accel to further increase efficiency.
- Extend the Either system to handle domains that involve relations among objects.

1993: - Evaluate Accel's ability to efficiently construct correct explanations in a NASA domain, possibly diagnosis of for a control system in the space shuttle.

- Continue evaluation and improvement of Accel's diagnostic ability, incorporating methods from Either to refine an initial knowledge base.

1994: - Begin adapting the combined Accel/Either system to a specific NASA diagnostic problem.

1995: - Begin systematic comparison of Accel's diagnostic behavior to that of existing systems for the selected domain.

4.5.7 Representation in Incremental Learning

Paul Utgoff; University of Massachusetts, Amherst

This work examines the influence of representation in inductive approaches to machine learning. Different representational formalisms have different biases, making some concepts easier to acquire in one formalism than in another. For instance, perceptrons (one-layer neural networks) can only learn to distinguish linearly separable concepts, whereas decision trees can make more complex types of discriminations. However, decision trees can only divide the space of instances in specific ways, which necessitates complex descriptions (and many training instances) for some concepts that are easy to represent and learn using perceptrons.

One thrust of Utgoff's research effort centers on developing hybrid formalisms, which combine desirable features from two or more representational schemes. For example, his previous work on perceptron trees shows that one can augment each terminal node of a decision tree with a perceptron. Experiments suggest that, in many cases, this hybrid approach gives a simpler concept description and more rapid learning.

In 1991, Utgoff extended this approach to allow perceptrons at internal nodes of induced decision trees. Tests on a pixel classification task (similar to that required for landsat data) showed that this approach produced simpler descriptions that were more accurate that ones learned by a traditional decision-tree algorithm. Research in 1992 will further refine and evaluate this approach to induction. In the longer term, Utgoff plans to augment this technique with the ability to incorporate formalisms besides decision trees and perceptrons. He also plans to explore approaches to representation change, in which the learner incrementally alters its descriptive language with experience.

Another focus of Utgoff's research involves adapting these approaches to induction, which were designed for simple classification tasks, for use in problem solving. Starting in 1992, he plans to employ a state-space paradigm in which one must decide which states to expand at each stage of the search process. As in Laird's work on Soar and Carbonell's work on PRODIGY, he plans to encode decision knowledge in terms of preference rules that give some states precedence over others. However, rather than using explanation-based methods to learn this search-control knowledge, the system would acquire it using incremental induction techniques, like those in his earlier work on decision trees and perceptron trees. Preliminary studies will focus on well-defined problem-solving tasks commonly used by the AI research community, but this framework could be applied to any task that one can view in terms of state-space search. These include scheduling problems for the Hubble Space Telescope and for Shuttle launch processing.

Milestones

1991: - Extend the perceptron tree method and evaluate its learning ability on classification tasks.

1992: - Complete initial experiments with perceptron-tree method, and extend framework to incorporate other techniques.

- Adapt the perceptron-tree method to acquire preference rules for state-space search; begin evaluation of approach.

1993: - Continue evaluation of approaches to learning preference, rules and begin adapting it to a NASA domain involving scheduling or control.

- Initiate research on representation change.

1994: - Evaluate and extend approaches to representation change, incorporating them into methods that learn preference rules for controlling search.
- Continue adaptation of preference-rule techniques to NASA domain.

1995: - Begin systematic evaluation of methods for learning preference rules in a NASA domain involving scheduling or control.

5 Knowledge Base Technology

5.1 Large-Scale Knowledge Bases

5.1.1 Multi-Use Knowledge Bases

Edward Feigenbaum, Yumi Iwasaki, Tom Gruber, Stanford University

The objective of this project is to develop a methodology and computational framework that will enable engineers to access a comprehensive body of engineering knowledge and reasoning methods in support of design verification, diagnosis, and interactive documentation. We expect that this comprehensive body of knowledge and methods will be reusable and thus reduce the knowledge engineering effort required for systems that must model the behavior of a physical device.

This project is split into a device modeling effort and a design knowledge capture effort. The device modeling issues addressed here include the, integration of quantitative and qualitative modeling, modeling at varying levels of abstraction, and integrating process modeling with component modeling. Qualitative simulation is useful when the complexity of the system prohibits or complicates the development of a mathematical model of the system. Also, qualitative representations could yield performance improvements over quantitative techniques because a substantial amount of detail is ignored. However, quantitative models are often required to help disambiguate the alternative branches in a qualitative simulation. The hope is that well-understood parts of the complex system could be modeled quantitatively but the majority of the model could remain qualitative.

Every device model incorporates simplifying assumptions depending on the modeling goal. Multiple models are usually built that do not explicitly refer to these goals. This project concentrates on capturing these goals so that models can be adequately described. Design goals usually refer to the following types of information:

- the types of questions that will be asked of the model like determining stability, comparative statics, or tracking dynamic transient behavior;
- the types of phenomena that are modeled like electrical, thermodynamic, structural, or kinetic properties;
- the model precision required including what orders of magnitude change in variable values are negligible or significant;
- the temporal grain size such as changes from second to second or week to week;
- the temporal scope which is the length of time that the model covers.

Because both processes and components need to be modeled to support multiple levels of abstraction, this project is exploring the integration of process simulation techniques and component simulation techniques.

As stated previously, design knowledge capture is also a goal of this project. The intent of this research is to highlight the crucial design decisions of a system and annotate designs with a justification of the design choices selected. These annotations will be acquired from designers with the help of a knowledge capture tool that is tightly integrated with the modeling tools discussed above.

To maximize realism and future technology transfer, this research is driven by the study of actual spacecraft subsystems. This year, a subset of the Electrical Power System (EPS) of the Hubble

Space Telescope was modeled. The project team concentrated on the nickel-cadium battery of the EPS.

This modeling was performed within the initial architecture of a Device Modeling Environment (DME) that supports simulation. The bulk of the year was spent on investigating alternative representation environments for the DME and developing DME prototypes. CYC, Hyperclass, Loom, and other public domain were evaluated. After building a preliminary DME in KEE the project team chose CYC because of its rich representation language. After porting CYC to the Stanford hardware/software environment, a new DME prototype was developed and used to simulate a simplification of the EPS charging and discharging cycles. Unfortunately, this exercise elucidated some weaknesses within the CYC implementation that resulted in rejecting CYC as the basis for the DME. CYC was a very large system that has not been "bullet-proofed" yet and this caused the team to spend more time manipulating CYC than investigating models. In the remaining portion of1991, the project team will develop a new DME in LISP using CLOS and a logical rule system. The hope is that this will evolve into a full-fledged DME with excellent robustness and power. The EPS will also drive this effort.

An implementation of a knowledge capture prototype will also be initiated in 1991. This system will allow designers to capture rationale by demonstration. The designer will identify the relevant portions of the device along with a model and then simulate the model with the DME. When the desired state is reached, the system will generate explanations. Then alternative conditions will be chosen, simulated, and finally explained. A tool that facilitates this process will be implemented and integrated with DME.

The project team plans to model a few devices at varying degrees of abstraction using the new DME and the knowledge capture tool. The EPS system will be modeled at varying levels of detail and other NASA systems will be explored. It is likely that an Ames-Rockwell collaboration in learning diagnostic systems will provide interesting models for the Stanford team to encode within the DME to test its efficiency, coverage, and ease of use. The Space Shuttle Fuel Cells are expected to be the first system explored by this collaboration and all data and knowledge will be sent to Stanford as another driving force behind the DME. The team is also considering the integration of the DME with a diagnostic performance engine.

Also in1992, the first design knowledge capture tool will be completed. Design knowledge capture experiments will be performed to evaluate the utility of the knowledge capture tool. NASA Ames and/or Stanford designers will be involved in these experiments.

Milestones

1991: - DME/CYC prototype completed.
- DME reimplemented in LISP.

1992: - Multiple NASA subsystems modeled within DME with ability shown to synthesize specialized rules for several purposes (design, diagnosis, scheduling, etc.).
- Design Knowledge Capture tool complete.

1993: - Complete prototype of large-scale engineering knowledge base.

5.2 Knowledge Acquisition and Use During Design

Design Acquisiton and Reuse

Catherine Baudin, Jody Gevins;
Larry Leifer, Ade Mabogunje, Venad Baya; Stanford Center for Design Research

The goal of this research is to develop methods and tools to capture the design evolution of an engineered device. We are developing a framework to represent design records that could facilitate the process of redesign or improve the design of similar devices.

This research is a collaboration between the Artificial Intelligence Research Branch and the Center for Design Research (CDR) at Stanford University.

We developed a prototype that assists in documenting the history of design modifications by comparing how a device meets the design requirements before and after the changes. The current system integrates the Electronic Design Notebook developed at the Center for Design Research with a knowledge-based system which can handle formal representations of the device structure and behavior and which is associated with a language to represent equations, design requirements and constraints. Each document of the notebook represents a design state and is associated with a formal model of the device being designed. The structure of the device and the design requirements are represented in a constraint network: a constraint manager monitors the interactions between design decisions and the device model. The system has a requirement analysis and a comparison component that explains design changes by evaluating their impact on the design constraints.

The Center for Design Research (CDR) developed the "Electronic Notebook Navigator" which allows VMACS documents to be indexed and browsed based on text strings associated with the notebook pages. The VMACS visual language is used as a front end to the navigator query language. We extended the indexing and browsing mechanism with a knowledge-based component that can hold links among the notebook pages. Relations among the pages are represented by MRS predicates. Different states of the design model can now be organized in a network and indexed by the requirement names or by results from the requirement analysis (constraints violated or satisfied). This enables a user to browse documents according to criteria such as the satisfaction of the design requirements, the type of analysis program that was invoked at a particular design stage or the type of changes that relate the design to a previous design state.

In a separate activity, Ken Shimizu and George Toye from CDR conducted an experiment that evaluated Failure Mode and Effect Analysis (FMEA) during the conceptual stages of a design, using qualitative simulation to perform the FMEA. Their experiment resulted in the implementation of an FMEA prototype that uses QSIM. Experimentation with this prototype indicated that the that this approach of limited utility because the QSIM envisionments were too large.

Finally, CDR students provided feedback on the clarity of our modeling language and on the quality of the explanations of design changes provided by the design comparison component. This provided us with a non-computer scientist point of view. As a result, we found that most explanations were difficult to understand because the system did not have any formal representation of a design change other than its impact on the design requirements and constraints. To alleviate this problem, we added the capability to compare the structure of two designs and to report on differences. Parameter changes and component changes are examples of these differences. In additon, we extended our system with CAD features such as advice generation that suggest design fixes to satisfy outstanding constraint violations. This advice is generated from the causal model which represents the interactions among the device components. In the present version, only parametric modifications can be suggested (i.e., which parameter value could be increased or decreased).

We will continue the evaluation of our approach to design knowledge capture by exploring design reuse issues. In particular, we will focus on the use of design rationale and of other elements of the design history to assist a user in performing design modifications. This will orient the research toward the use of design traces for computer assisted redesign -- a form of derivational analogy. By redesign, we mean the task of finding what aspect of a design should be modified and what decisions should be revised to accomodate a change in design specifications. In an earlier experiment, we modeled part of the conceptual stages of the SIRTF telescope; we are now starting a new experiment with the Stanford Center for Design Research. This new experiment will capture the design evolution of a small mechanical system and will cover conceptual design stages as well as more detailed steps of the process. The goal of the study is to find out what information should be recorded during the design process of a mechanical device to facilitate later redesign of this device or the design of similar devices. Our goal is to show that the system can help a designer manage the ramifications of a design change by pointing out the design decisions that could be revised and the parts of the design that can remain unchanged. Ade Mabogunje and Vinod Baya will be acting as our mechanical design experts and will solve a design case selected from a Stanford mechanical engineering class. An informal trace of the design will be captured on videotape and/or VMACS text documents. We will experiment with alternative representations of the design trace and evaluate whether a designer would benefit from having this representation available as reference material. At the end of our one year experiment we should be able to evaluate how well our current approach addresses the following two questions:

- What part of the design process is useful for later redesign?

- How should this design information be represented for computer assisted design reuse.

Milestones

1991: - FMEA - QSIM prototype completed.
- A new NASA design problem chosen.
- Redesign experiment initiated.
- Conduct NASA designer's workshop on use of VMACS tools.

1992: - Redesign experiment completed.
- Demonstration of use of design notebook tools for comparative analysis of alternative designs.

1993: - Automatic design trace retrieval mechanism incorporated into VMACS.
- Field use of Electronic Design Notebook by NASA engineers.

1994: - Completion of initial integration of Electronic Design Notebook with CAD/CAM tool.

5.2.2 Learning and qualitative reasoning in diagnosis

Deepak Kulkarni, Peter Robinson

This work seeks to show how the performance of a FDIR (failure detection, isolation, and recovery) system with a fault tree can be improved using learning and qualitative reasoning techniques. Observations of the behavior of a physical system consist of the readings of sensors at a certain sampling rate. Usually available knowledge allows inference of faults from a semi-qualitative description, and not from raw data from sensors. Therefore a fault diagnosis system needs to extract the qualitative features from the sensor readings. In some cases, the features are hills and valleys on the top of a complex baseline. In some other cases, the features are qualitative values such as 'high' and 'low' that are acquired by comparing sensor readings with predefined thresholds.

We have developed procedures to extract hills and valleys in the data using machine vision techniques. In 1991, we are developing mechanisms to deal with dynamic changes in what is considered a high or a low value. We are also using qualitative reasoning to detect missing branches in an expert-specified fault tree.

A number of learning algorithms (e.g. ID3) allow us to induce a tree from a set of failure examples. Some work has also been done on guiding the induction using pre-existing knowledge (e.g., KAISER). We plan to use the expert-specified tree to guide the induction process. The fault tree produced by the learning algorithm would cover more failure conditions than the original tree. Learning techniques can also generate abstract fault classes, that are useful for detecting novel faults. In 1990, we demonstrated this for a simplified heating device. In 1991, we will identify a NASA domain for application of these techniques; our current plan is to work on a Shuttle fuel cell diagnosis system in collaboration with Rockwell in Downey, California. We will attempt to show that the learning algorithm will provide broader coverage than the fault-tree in the Shuttle fuel-cell domain.

AI researchers have developed techniques for simulating qualitative models. However models for many devices include both qualitative relationships and quantitative equations. We plan to develop a simulator that can use both qualitative and quantitative parameter information. Qualitative reasoning can be used to improve the performance of a FDIR system in a number of ways. First, it can be used to generate instances of behavior for a learning algorithm. Secondly, it can be used to envision alternate qualitative states that will follow a particular qualitative state. That will allow detection of missing branches in a fault tree. Finally, it will be used to verify isolation and recovery procedures before performing the actions on actual hardware.

1991: - Begin work on Shuttle fuel cell diagnosis domain.
- Develop a learning mechanism that can use a pre-defined fault tree in guiding induction from examples.

1992: - Develop an algorithm that can use qualitative reasoning to detect missing branches in a fault tree.
- Develop a mechanism that uses qualitative reasoning to test isolation and recovery procedures.

1993: - Demonstrate the utility of the techniques for a significant size NASA task domain.

5.2.3 AI and Multi-Faceted Modeling

Bernard Zeigler, University of Arizona

Note that this task is projected to end in 1991. Therefore, we provide no 1992 plans nor further milestones.

The purpose of this work is to develop concepts and tools for multi-abstraction models that can be used for a variety of purposes including design, diagnosis and operations. In previous work, we focussed on elaborating the theory of event-based control of devices and developing the tools neccessary to implement it. An autonomous, intelligent agent could in principle, base its operation, diagnosis, repair, planning, and other reasoning activities on a single comprehensive model of its environment. However, such a model would be extremely unwieldly to develop and lead to intractable computations in practice. Instead, we proposed an architecture for autonomous systems in which a multiplicity of partial models supports system objectives. We have demonstrated how the partial models can be easier to develop and more computationally tractable. Since this approach leads to sets of overlapping and redundant representations expressed in various formalisms and levels of abstraction, we proposed concepts that organize such representations into a coherent architecture including: event-based generic control engines, abstraction hierarchies, and their association with matching hierarchical control structures. These are being implemented within DEVS-Scheme, our knowledge-based simulation environment for hierarchical, modular modeling and simulation.

We have developed automatic means to map models of devices into abstractions that can be used for operation and diagnosis. The significance is that knowledge available from the design of a device or instrument can be more readily converted into a) a model that can be used to operate the device and b) a model that can be used to do fault diagnosis should an anomaly in operation arise. The abstraction of operational models has been implemented.

We have extended our DEVS-Scheme simulation environment to handle symbolic simulation that can be used to generate multiple fault tables for multicomponent models. In this approach, we develop fault models that represent possible abnormal consequences of elementary robot actions. These are propagated in all possible combinations (subject to constraints) using symbolic simulation to generate observable abnomalies. We have established the need for including fault models in the system description for trouble shooting using Reiter's theory of diagnosis from first principles. Our generation of fault tables extends existing work in that it explicitly incorporates timing and dynamic behavior that arises naturally from the discrete event formalism.

We have also added the capability to our DEVS-Scheme simulation environment to function in an intelligent, real-time control mode. The significance is that event-based, intelligent control systems that are developed in our existing simulation mode can now be interfaced directly to run as actual, real-time controllers. We have demonstrated the application of this approach to an actual prototype implementation of an oxygen extraction plant being developed by the University of Arizona-NASA Space Engineering Center from which we also have research support.

Tools Available For NASA Use

Our work has resulted in further extensions of the DEVS-Scheme knowledge-based simulation environment as described above. This environment is available for NASA use and is being commercialized with seed money from the University of Arizona Technology Development Center. Some features that should be applicable to NASA design of high autonomy systems are:

1. Hierarchical, modular discrete event simulation model development
2. Supports knowledge-based system design by reusable model bases
3. Supports automated model abstraction of devices for operation and diagnosis
4. Enables symbolic simulation for multiple fault analysis
5. Facilitates migration of event-based, intelligent controllers designed with simulation to real-time operation

Applications

The University of Arizona-NASA Space Engineering Center is constructing a demonstration of how in-situ propellant production on Mars would work using a zirconia-based oxygen production cell concept (converting Mars atmosphere rich in carbon dioxide to oxygen. The AI-Simulation Research Group is taking a leading role in the design of the demonstration system and its control. The project serves as an opportunity to apply the concepts and techniques for design of high autonomy systems being developed with support of the NASA-Ames AI Research Branch.

The University of Arizona (Colleges of Engineering, Agriculture and Environmental Research Laboratory) proposed a NASA Center for Bioregenerative Life Support Systems. The objective is to develop the knowledge base necessary to establish bioregenerative life support systems on the Moon or other planets. To achieve this objective a test-bed facility was proposed in which various alternatives for bioregenerative self-sustaining systems could be studied. The AI-Simulation group provided the system entity structuring for the design of this test-bed. If funded, the group will be at the center of the integration of the multi-disciplinary effort and will also be responsible for the intelligent control and automation of the test-bed.

BIBLIOGRAPHY

Journals, Books, and Major Peer-Reviewed Conferences

H.R. Berenji, Y.Y. Chen, C.C. Lee J.S. Jang, and S. Murugesan. A Hierarchical Approach to Designing Approximate Reasoning-Based Controllers for Dynamic Physical Systems. In P.P. Bonissone, M. Henrion, L.N.Kanal, and J. Lemmer, editors, Uncertainty in Aritficial Intelligence: Volume VI, in the series Machine Intelligence and Pattern Recognition, Elsevier, North-Holland, 1991, (in press).

H.R. Berenji. Fuzzy Logic Controller. In R.R. Yager and L.A. Zadeh, editor, An Introduction to Fuzzy Logic Applications in Intelligent Systems, Kluwer Academic Publishers, 1991, (to appear).

H.R. Berenji. An Architecture for Designing Fuzzy Controllers using Neural Networks. International Journal of Approximate Reasoning, 1991 (in press).

H.R. Berenji, Y.Y. Chen, and R.R. Yager. Using New Aggregation Operators in Rule-Based Intelligent Control. In 29th IEEE Conference on Decision and Control, pages 2198-2203, Honolulu, Hawaii, 1990.

H.R. Berenji, Y.Y. Chen, C.C. Lee, S. Murugesan, and J.S. Jang. An Experiment-Based Comparative Study of Fuzzy Logic Control. In American Control Conference, Pittsburgh, 1989.

G.A. Boy. Intelligent Assistant Systems. Academic Press, London, 1991.

G.A. Boy and N. Mathe, (1989). Operator Assistant Systems: An Experimental Approach Using a Telerobotics Application. IJCAI Workshop on Integrated Human-Machine Intelligence in Aerospace Systems, Detroit, Michigan, USA, August 21; and International Journal of Intelligent Systems, Special Issue on Knowledge Acquisition (1991), Ken Ford and Jeff Bradshaw (Eds.).

G.A. Boy, (1990). Acquiring and Refining Indices According to Context. Proceedings of the Fifth AAAI-Sponsored Knowledge Acquisition for Knowledge-Based Systems Workshop, Banff, Canada, November 4-9. To appear in the Knowledge Acquisition Journal.

W.L. Buntine and D.A. Stirling. Interactive Induction. In J. Hayes, D. Michie and E. Tyugu, Editors, MI-12: Machine Intelligence 12, Machine Analysis and Sythesis of Knowledge, Oxford University Press, Oxford 1990.

W.L. Buntine. Myths and Legends in Learning Classification Rules. In Eighth National Conference on Artificial Intelligence, pages 736-742, Boston, Massachuseets, 1990.

W.L. Buntine. Modelling Default and Likelihood Reasoning as Probabilistic Reasoning. Annals of Mathematics and AI, 1991, (in press).

W.L. Buntine and T. Niblett. A Further Comparison of Splitting Rules for Decision-Tree Induction. Machine Learning, 1991, (in press).

W.L. Buntine. Learning Classification Trees. Statistics and Computing. (in press).

Cheeseman, J. Goebel, K. Volk, F. Gerbault, M. Self, J. Stutz and W. Taylor. A Bayesian Classificationof the IRAS LRS Atlas. Astronomy and Astrophysics. 222, L5-L8 (1989).

P. Cheeseman, (July 1990). On the Importance of Evidence: A Response to Halpem. Proceedings in the Computational Intelligence Journal.

P. Cheeseman, B. Kanefsky and W. Taylor. "Where the Really Hard Problems Are". to appear in Proceedings of IJCAI-91.

P. Cheeseman,R. Hanson and J. Stutz. "Bayesian Classification with Correlation and Inheritance". to appear in Proceedings of IJCAI-91.

S.P. Colombano, L.R. Young, G. Haymann-Haber, N. Groleau, P. Szolovits and D. Rosenthal. "An Expert System to Advise Astronauts During Experiments". 40th Congress of the International Astronautical Federation, Oct. 1989.

M. Drummond, A. Tate and J. Hendler. AI Planning: Systems and Techniques. AI Magazine, Vol. 11, No. 2, pp. 61--77 (Summer 1990). Translated into Japenese and reprinted in Nikkei Artificial Intelligence (Extra Publication) Autumn 1990, pp. 108-125

M. Drummond and J. Bresina. Anytime Synthetic Projection: Maximizing the Probability of Goal Satisfaction. In proc. of AAAI-90.

M. Drummond, A. Tate and J. Hendler. A Review of AI Planning Techniques. In 'Readings in Planning", published by Morgan-Kaufmann, San Mateo, CA pp. 26--49, 1990.

M. Drummond, S. Minton and J. Bressina. Commitment Strategies in Planning: A Comparative Analysis. to appear in Proceedings of IJCAI-91.

Eskey, M., and Zweben, M., "Learning Search Control for Constraint-Based Scheduling", Proceedings of the National Conference on Artificial Intelligence, 1990.

E.A. Feigenbaum, Y. Iwasaki and R.M. Keller. "A Response to Chandrasekaran's Viewpoint on GenericTasks", Knowledge Engineering Review, 1989.

S. Kedar and S. Kambhampati. "Explanation-Based Generalization of Partially-Ordered Plans. to appear in Proceedings of AAAI-91. Anaheim, CA.

R.M. Keller , C. Baudin, Y. Iwasaki, P. Nayak and K. Tanaka. "Compiling Redesign Plans and DiagnosisRules from a Structure/Behavior Device Model", to appear in Knowledge Aided Design. M. Green(Ed.), Academic Press, London, 1991.

R.M. Keller. "Applying Knowledge Compilation Techniques to Model-based Reasoning", to appear in IEEE Expert, 1991.

D. Kulkarni and H.A. Simon, (1990). "Experimentation in Machine Discovery." In P. Langley & J. Shrager (Eds.) Computational Models of Discovery and Theory Form.

P.Laird. A survey of computational learning theory. In R. Banerji, editor, Formal Techniques in Artificial Intelligence: A Sourcebook., Elsevier Science, 1990.

P. Laird. Extending EBG to term-rewriting systems. In Proceedings, AAAI-90, pages 929 -- 935, Morgan Kaufmann, Menlo Park, 1990.

P. Laird and E. Gamble. Analytical learning and term-rewriting systems. 1990. Submitted to Artificial Intelligence Journal.

P. Laird and E. Gamble. A PAC Algorithm for Making Feature Maps. Machine Learning 5 (1), 1991.

P. Langley and K.B. McKusick, (1990). "Constraints on Tree Structure in Concept Formation." Twelfth International Joint Conference on Artificial Intelligence.

P. Langley and J.A. Allen, (1991). "Inductive Acquisition of Plan Knowledge." Ninth National Conferenceon Artificial Intelligence.

P. Langley, K. Thompson and W.F. Iba, (1991). "The Utitlity of Background Knowledge in Concept Formation." Ninth National Conference on Aritificial Intelligence.

P. Langley and B. Nordhausen, (1990). "An Integrated Approach to Empirical Discovery." In J. Shrager and P. Langley (Eds.). Computational Models of Scientific Discovery and Theory Formation. San Mateo,CA: Morgan Kaufmann.

P. Langley and D.H. Fisher, (1990). "The Structure and Formation of Natural Categories." In G.H. Bower(Eds.). The Psychology of Learning and Motivation: Advances in Research and Theory. Vol.-26. Cambridge, MA: Academic Press.

P. Langley and J. Shrager, (1990). Computational Approaches to Scientific Discovery. In J. Shrager and P. Langley (Eds.). Computational Models of Scientific Discovery and Theory Formation. San Mateo,CA: Morgan Kaufmann.

P. Langley and K. Thompson, (In press). "Concept Formation in Structured Domains." In D. Fisher and M. Pazzani (Eds.). Computational Approaches to Concept Formation. San Mateo, CA: Morgan Kaufmann.

P. Langley and J.C. Schlimmer, (In press). "Machine Learning". In S. Shapir (Eds.). Encyclopedia of Artificial Intelligence (2nd ed.). New York: John Wiley & Sons.

P. Langley and J.A. Allen, (In press). "A Unified Framework for Planning and Learning." In S. Minton (Eds.). Learning to Plan. San Mateo, CA: Morgan Kaufmann.

P. Langley, (1990). Approaches to Learning and Representation. Behavioral and Brain Sciences.

P. Langley, and J. Shrager, (1990). Computational Models of Scientific Discovery and Theory Formation. San Mateo, CA: Morgan Kaufmann.

W. Iba, (1990). "Learning to Classify Observed Motor Behavior". to appear in Proceedings of IJCAI-91.

W. Iba and J. Gennari. "Learning to Recognize Movements". (In press - - book chapter). In D. Fisherand M. Pazzani (Eds.). Computational Approaches to Concept Formation. San Mateo, CA: Morgan Kaufmann.

A. L. Lansky. "Localized Search for Multiagent Planning". to appear in Proceedings of IJCAI-91.

A.L. Lansky. "Localized Representation and Planning", in Readings in Planning, J. Allen, J. Hendler, and A. Tate (editors), Morgan Kaufman Publishers, pp. 670-674 (1990).

S. Minton, M.D. Johnston, A.B. Philips and P. Laird. Solving Large-Scale Constraint-Satisfaction and Scheduling Problems Using a Heuristic Repair Method. Proceedings of the National Conference on Artificial Intelligence, 1990.

S. Minton, J.G. Carbonell, Y. Gil, R. Joseph, C.A. Knoblock and M.M. Veloso. Designing an Integrated Architecture: The Prodigy View. Proceedings of the Twelfth Annual Conference of the Cognitive Science Society, 1990.

S. Minton, C. Knoblock and O. Etzioni. "Integrating Abstraction and Explanation-Based Learning in Prodigy". to appear in Proceedings of the National Conference on Arificial Intelligence, 1991.

S. Minton, J. Bresina and M. Drummond. "Commitment Strategies in Planning: A Comparative Analysis, " to appear in Proceedings of IJCAI-91.

S. Minton. "Quantitative Results Concerning the Utility of Explanation-Based Learning" Artificial Intelligence, Vol. 42, 1990. Also reprinted in "Readingsin Machine Learning ", Jude W. Shavlik and Thomas G. Dietterich (Eds.), Morgan Kaufmann Publishers,San Mateo, CA, 1990.

S. Minton, J.G. Carbonell and C. Knoblock. "Prodigy: An Integrated Architecture for Planning and Learning", in Architectures for Intelligence, K. VanLehn (Eds.), Erlbaum, 1990.

S. Minton, J. Carbonell, C. Knoblock, D.R. Kuokka, O. Etzioni and Y. Gil. "Explanation-Based Learning:A Problem-Solving Perspective", Artificial Intelligence, Vol. 40, Sept. 1989. Also reprinted in Machine Learning: Paradigms and Methods, J.G. Carbonell (Eds.), MIT Press, 1990.

M. Zweben. "Space Shuttle Processing: A. Case Study in Artificial Intelligence", 28th Space Congress, 1991.

Conference/Workshop Proceedings/Technical Reports

C. Baudin, C. Sivard and M. Zweben. "A Model Based Approach to Design Rationale Conservation". In Proceedings of AAAI workshop on Model Based Reasoning 1989.

C. Baudin, C. Sivard and M. Zweben. "Using Device Models and Design Goals for Design Rationale Capture". The AAAI workshop on explanation - Boston, MA , July 30, 1990.

C. Baudin, C. Sivard and M. Zweben. "Recovering rationale for Design Changes: A Knowledge-Based Approach". IEEE 1990, International conference on Systems, Man and Cybernetics

H.R. Berenji. Neural Networks and Fuzzy Logic in Intelligent Control. In Fifth IEEE Internation Symposiumon Intelligent Control, pages 916-920, Philadelphia, PA 1990.

H.R. Berenji. Machine Learning in Fuzzy Control. In Int. Conf. on Fuzzy Logic & Neural Networks, pages 231-234, Iizuka, Fukuoka, Japan, 1990.

H.R. Berenji. An Architecture for Designing Fuzzy Controllers Using Neural Networks. In Second Joint Technology Workshop on Neural Networks and Fuzzy Logic, Houston, Texas, April 1990.

H.R. Berenji, Y.Y. Chen, C.C. Lee, J.S. Jang and S. Murugesan. A Hierarchical Approach to Designing Approximate Reasoning-Based Controllers for Dynamic Physical Systems. in Sixth Conference on Uncertainty in Artificial Intelligence, pages 362-369, 1990.

W.L. Buntine. Learning Classification Trees. Technical Report FIA-90-12--19-01, RIACS and NASA . Paper presented at Third International Workshop on Artificial Intelligence and Statistics. Submitted,IJCAI-91.

G.A. Boy, (1989). Interactive Knowledge Acquisition for Intelligent Documentation. Proceedings of the AAAI-89 Workshop on Knowledge Acquisition: Practical Tools and Technicques, Detroit, MI, August 23.

G.A. Boy, (1989). The Block Representation in Knowledge Acquisition for Computer Integrated Documentation. Proceedings of the Fourth AAAI-Sponsored Knowledge Acquisition for Knowledge-Based Systems Workshop, Banff, Canada, October 1-6.

G.A. Boy, (1989). Computer Integrated Documenation: A Problem of Knowledge Acquisition and Representation. Proceedings of the ESA-ESTEC Workshop on Human Factors Engineering: A Task Oriented Approach, Noordwijk, The Netherlands, November 21-23.

G.A. Boy and T. Gruber, (1990). Intelligent Assistant Systems: Support for Integrated Human-Machine Systems. Proceedings of the AAAI Spring Symposium on Knowledge-Based Human Computer Communication, Stanford, March 27-29.

G.A. Boy, (1990). Acquiring and Refining Procedure According to Context. Proceedings of the AAAI-90 Workshop on Knowledge Acquistion: Practical Tools and Techniques, Boston, MA, July 29.

G.A. Boy, (1990). Advanced Interaction Media. Proceedings of the Third Conference on Human-Machine Interaction and Artificial Intelligence in Aeronautics and Space. Toulouse, France, September 26-28.

P. Cheeseman and B. Kanefsky, (March 1990). Evolutionary Tree Reconstruction. Proceedings of the American Association ofArtificial Intelligence (AAAI) Spring Symposium on "Minimum Message Length Encoding". Spring 1990, Stanford University.

S.P. Colombano, G. Haymann-Haber, N. Groleau, D. Rosenthal, P. Szolovits and L.R. Young.

An Expert System to Advise Astronauts During Experiments: The Protocol Manager Module". Proceedings of the Conference on Space Automation and Robotics, July 1989.

S.P. Colombano, R. Frainier, N. Groleau, R. Bhatnager, C. Lam, M. Compton, S. Lai, P. Szolovits, M. Manahan, I. Statler and L. Young. A Comparison of CLIPS- and LISP Based Approaches to the Development of a Real-Time Expert System. Proceedings of the First CLIPS Users Conference, August 1990.

M. Drummond and J. Bresina. Integrating Planning and Reaction: A Preliminary Report. Proceedings of the AAAI Spring Symposium Series (session on Planning in Uncertain, Unpredictible, or Changing Environments), 1990.

M. Drummond, J. Bresina and S. Kedar. The Impact of Learning on an Integrated Architecture. AAAI-91 Stanford Spring Symposium.

M. Drummond and J. Bresina. Planning for Control. In proc. of Fiftth IEEE International Sysmposium on Intelligent Control, published by the IEEE Computer Society Press, Philadelphia, PA. (pp. 657--662., 1990).

M. Drummond and L. Kaelbling. Integrated Agent Architectures: Benchmark Tasks and Evaluation Metrics. In proc. of DARPA Workshop on Innovative Approaches to Planning, Scheduling and Control, San Diego, CA. (pp. 408--411). Morgan Kaufmann, San Mateo, CA. 1990.

M. Drummond and P. Langley. Toward an Experimental Science of Planning. In proc. of DARPA Workshop on Innovative Approaches to Planning, scheduling and Control, San Diego, CA. (pp. 109--114). Morgan-Kaufmann, San Mateo, CA 1990.

N. Groleau, R. Bhatnager and D.M. Merfeld. Using Qualitative Knowledge for Quantitative Simulationof the Human Spatial Orientation System. Proceedings of the Second Annual Conference on AI, Simulation and Planning in High Autonomy Systems, April 1991.

S. Kedar, J. Bresina and M. Drummond. "The Impact of Learning on an Integrated Architecture." AAAI Spring Symposium on Integrated Intelligent Architectures. March, 1991. Stanford, CA.

S. Kedar and M. Lowry. Explanation-Based Generalization Applied to Software Reengineering. AAAI-91 Workshop on Automating Software Design, July, 1991, Anaheim, CA.

S. Kedar, J. Bresina and L. Dent. "Refining Mutually Incomplete Knowledge Sources Through Failure." submitted to the Eighth International Workshop on Machine Learning. June, 1991.

R.M. Keller, M. H. Sims, E. Podolak and C.P. McKay. "Constructing an Advanced Software Tool forPlanetary Atmospheric Modeling", Proc. 1990 International Symposium on Artificial Intelligence, Robotics and Automation in Space, Kobe, Japan, November 1990

R.M. Keller and J.L. Dungan. Development of an Advanced Software Tool for Ecosystem Simulation Modelling, abstract submitted to Ecological Society of America Conference, 1991.

R.M. Keller. "In Defense of Compilation: A Response to Davis' 'Form and Content in Model-Based Reasoning' ", Proc. 1990 Workshop on Model-Based Reasoning, Boston, MA, June 1990.

R.M. Keller, C. Baudin, Y. Ywasaki, P. Nayak and K. Tanaka. Model Compilation: An Approach to Automated Model Derivation, Proc. Change of Representation and Problem Reformulation Workshop, Palo Alto, CA, March 1990.

R.M. Keller, C. Baudin, Y. Ywasaki, P. Nayak and K. Tanaka. Deriving Shallow Rules from Underlying Device Models, Proc. 1989 Workshop on Model-Based Reasoning, Detroit, MI, August 1989.

R. M. Keller, Y. Iwasaki, K. Doshi, T. Gruber and C.M. Low. Equation Model Generation: Where do equations come from?, Proc. 1989 Workshop on Model-Based Reasoning, Detroit, MI, August 1989.

R.M. Keller. Compiling Learning Vocabulary from a Perfomance System Description, Proc. Sixth International Workshop on Machine Learning, Ithaca, NY, June 1989.

R.M. Keller, C. Baudin, y. Iwasaki, P. Nayak and K. Tanaka. Compiling Diagnosis Rules and Redesign Plans from a Structure/Behavior Device Model: The details, Knowledge Systems Lab, Technical Report No. KSL-89-50, Stanford, CA, June 1989.

D. Kulkarni and D. E. Thompson. Application of Scale-Space Filtering and LABYRINTH to Soil Analysis. In Workshop on Pattern Discovery In Large Data-Bases, 1990.

D. Kulkarni, D.E. Thompson, R. Mancinell, R. Levinson and P. Robinson. Autonomous Control and Hypothesis Generation for a Coupled Differential Thermal Analyzer - - Gas Chromatograph Instrument. In Pacific Conference in Chemistry and Spectroscopy, 1990.

D. Kulkarni and H.A. Simon, (1989). The Role of Experimentation in Scientific Theory Formation. In Proceedings of Sixth International Workshop on Machine Learning. (pp. 278-283). Ithaca.

P. Laird and E. Gamble. EBG and term-rewriting systems. In Proceedings of the First International Workshop on Algorithmic Learning Theory, pages 425 -- 440, Japanese Society for Artificial Intelligence, Tokyo, 1990.

P. Langley and B. Nordhausen, (1990). A Robust Approach to Numeric Discovery. Proceedings of the Seventh International Conference on Machine Learning. (pp.411 - 418). Austin, TX: Morgan Kaufmann.

P. Langley and J.A. Allen, (1990). Integrating Memory and Search in Planning. Proceedings of the 1990 DARPA Workshop on Innovative Approaches to Planning, Scheduling, and Control. (pp.301 - 312). San Diego, CA: Morgan Kaufmann.

P. Langley and M. Drummond, (In press). Toward an Experimental Science of Planning. Proceedings of the 1990 DARPA Workshop on Innovative Approaches to Planning, Scheduling, and Control. San Diego, CA: Morgan Kaufmann.

P. Langley, D. Billman, D. Fisher, M. Gluck and M. Pazzani, (1990). Computational Models of Category Learning. Proceedings of the Twelfth Conference of the Cognitive Science Society. (pp. 989 - 996).Cambridge, MA: Lawrence Erlbaum.

A. Lansky. Localized Search for Controlling Automated Reasoning. Proceedings of the 1990 DARPA Workshop on Innovative Approaches to Planning, Scheduling, and Control. San Diego, CA: Morgan Kaufmann Publishers. (pp. 115 - 125) (1990).

A. Lansky. Localized Search for Complex Planning Domains. Proceedings of 1990 AAAI Workshop on Planning for Complex Domains.

S. Minton and A.B. Phillips. Applying a Heuristic Repair Method to the HST Scheduling Problem. Proceedings of the DARPA Workshop on Innovative Approaches to Planning, Scheduling and Control. Edited by K. Sycara, San Diego, CA 1990.

S. Minton. Issues in the Design of Operator Composition Systems. Proceedings of the Seventh International Conference on Machine Learning, 1990.

A. Ralescu and H. R. Berenji. Integrating Structured Knowledge and Management of Uncertainty in Intelligent Systems. In International Joint Conference on Artificial Intelligence Workshop on Conceptual Graphs, Detroit, MI, 1989.

D.E. Thompson, et al, (1990). Autonomous Control and Hypothesis Generation of a Coupled Differential Thermal Analyzer - Gas Chromatograph Instrument. Invited Presentation, 1990. Pacific Conference on Chemistry and Spectroscopy, San Francisco. (pp. 48-49, Proceedings).

T.Whalen and H.R. Berenji. Actions as evidence: Multiple Epistemic Agents Acting Under Uncertainty. In Second IEEE Annual Conference on AI Simulation and Planning in High Automony Systems, Cocoa Beach, Florida, 1991.

M. Zweben. "Constraint-Based Simulated Annealing", AAAI-90 Workshop on Constraint-Directed Reasoning.

M. Zweben . "Anytime Rescheduling", DARPA Workshop on Innovative Approaches to Planning, Scheduling, and Control.